Universidad Zaragoza

Trabajo Fin de Grado

Grado en ingeniería informática

Un modelo predictivo basado en deep learning para la caracterización de la evolución de la enfermedad de Alzheimer

Autor

Ubaldo Ramón Júlvez

Directoras

Elvira Mayordomo Cámara

Mónica Hernández Giménez

Escuela de ingeniería y Arquitectura

2019

TRABAJOS DE FIN DE GRADO / FIN DE MÁSTER

Escuela de Ingeniería y Arquitectura Universidad Zaragoza

DECLARACIÓN DE AUTORÍA Y ORIGINALIDAD

(Este documento debe entregarse en la Secretaría de la EINA, dentro del plazo de depósito del TFG/TFM para su evaluación).

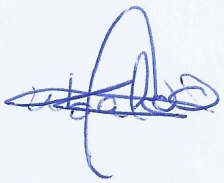
D./Dª. UBALDO RAMÓN JÚLVEZ ,en aplicación de lo dispuesto en el art. 14 (Derechos de autor) del Acuerdo de 11 de septiembre de 2014, del Consejo de Gobierno, por el que se aprueba el Reglamento de los TFG y TFM de la Universidad de Zaragoza,

Declaro que el presente Trabajo de Fin de (Grado/Máster) Grado en ingeniería informática (Título del Trabajo)

Un modelo predictivo basado en deep learning para la caracterización de la evolución de la enfermedad de Alzheimer

es de mi autoría y es original, no habiéndose utilizado fuente sin ser citada debidamente.

Zaragoza, 29 de Agosto de 2019



Fdo: UBALDO RAMÓN JÚLVEZ

RESUMEN

Un modelo predictivo basado en deep learning para la caracterización de la evolución de la enfermedad de Alzheimer

La enfermedad de Alzheimer es una enfermedad neurodegenerativa que afecta a más de 30 millones de personas en todo el mundo, lo que la convierte en una de las formas de demencia más comunes. Actualmente no se posee suficiente conocimiento ni de las causas de la enfermedad ni de su evolución como para realizar un diagnóstico pre-mortem concluyente. En la mayoría de los casos se puede realizar un diagnóstico de probable enfermedad de Alzheimer cuando es observable un deterioro cognitivo agudo, lo que indica que la enfermedad se encuentra en un estadio avanzado. Aunque no se conoce una cura para el alzhéimer, sí que existen tratamientos que pueden ralentizar el desarrollo de la enfermedad. Dichos tratamientos muestran más efectividad si se comienzan a aplicar durante las etapas más tempranas de la enfermedad. Por ello, existe un gran interés en la identificación de biomarcadores y en el desarrollo de sistemas predictivos útiles en el diagnóstico precoz de la enfermedad de Alzheimer.

El principal objetivo de este Trabajo de Fin de Grado es el desarrollo de un sistema basado en redes neuronales convolucionales profundas que, mediante el uso de biomarcadores clínicos, genéticos y de imagen sea capaz de predecir qué pacientes con deterioro cognitivo leve desarrollarán la enfermedad de Alzheimer en un plazo de tres años. El sistema también es capaz de diferenciar entre individuos sanos y enfermos con una precisión del 100%. Para el ajuste de los parámetros del sistema se han utilizado los datos de 778 individuos del repositorio de la *Alzheimer's Disease Neuroimaging Initiative* (ADNI).

El sistema desarrollado se ha implementado dentro de un marco de *multi-task learning* en el que la red neuronal convolucional es compartida por los dos clasificadores que se encargan de aprender la diferenciación entre individuos sanos y enfermos y de predecir la evolución del estado del individuo. La arquitectura de dicha red es la descrita en [1], siendo esta hasta la fecha el método que ha obtenido mejores resultados en los dos problemas propuestos. Las capas de la red combinan capas convolucionales 3D con capas convolucionales separables y otras capas más habituales en sistemas de *deep learning*. La implementación del sistema se ha realizado en Python 3.6, mediante el uso de las librerías de Keras y TensorFlow. El entrenamiento y test de la red se ha realizado sobre una tarjeta gráfica Titan RTX de 24 GBs de VRAM.

Los experimentos realizados muestran que se han conseguido replicar los resultados previamente obtenidos en [1]. Además se han realizado diversos experimentos con diferentes combinaciones de los datos de entrada para estudiar la sensibilidad del sistema a dichos cambios. Se ha encontrado que la mejor combinación de datos de entrada consiste en utilizar información clínica junto con las imágenes MRI. Los resultados obtenidos ponen en duda la necesidad de realizar la normalización de las imágenes utilizando registro no-rígido, como viene siendo necesario para otros sistemas de aprendizaje menos potentes. Finalmente, se han aplicado diferentes técnicas de análisis de redes neuronales convolucionales para identificar las limitaciones del modelo y proponer mejoras al sistema.

GLOSARIO

A continuación se presenta un glosario con las siglas de uso no general que se utilizan en este trabajo:

- ACC: *Accuracy*.
- AD: *Alzheimer's Disease*.
- ADNI: *Alzheimer's Disease Neuroimaging Initiative*, repositorio online de datos para su uso en investigaciones relacionadas con el alzheimer.
- ANTS: *Advanced Normalization ToolS*, librería con herramientas para el preprocesado de imágenes MRI. Incluye algoritmos para la corrección del bias field en imágenes MRI y ANTS-SSD, un algoritmo de registro con difeomorfismos.
- API: *Application programming interface.*
- AUC: *Area Under Curve*.
- BL PDE-LDDMM: *Band-Limited Partial Differential Equation constrained Large Deformation Diffeomorphic Metric Mapping*. Hace referencia a un algoritmo eficiente de registro con difeomorfismos.
- CNN: C*onvolutional Neural Network*.
- FC: *Fully Connected*.
- FOV: *Field Of View*.
- FMRIB: *Functional Magnetic Resonance Imaging of the Brain*.
- FSL: FMRIB *Software Library*, librería con herramientas para el preprocesado de imágenes MRI cerebrales.
- GPU: *Graphics Processing Unit.*
- HC: *Healthy Control* , pacientes sanos usados como controles en el estudio.
- MRI: *Magnetic Resonance Imaging/Image*, técnica de adquisición de imágenes médicas consistente en la aplicación de campos magnéticos para generar imágenes de los tejidos.
- MCI: *Mild Cognitive Impairment*, deterioro cognitivo leve.
- NCC: *Normalized Cross Correlation*, métrica de similitud entre imágenes ampliamente utilizada en los algoritmos de registro.
- PET: *Positron Emission Tomography.*
- pMCI: *progressive* MCI.
- ROC:*Receiver Operating Characteristic.*
- rs fMRI: *resting-state functional* MRI.
- SEN: *Sensitivity*.
- sMCI: stable MCI.
- SPE: *Specificity*.
- SNP: *Single Nucleotide Polymorphism*, mutación en la secuencia de ADN de un genoma que afecta a una sola base.
- SVM: *Support Vector Machine*.
- SSD: *Sum of Squared Differences*, métrica de similitud entre imágenes ampliamente utilizada en los algoritmos de registro.
- VRAM: *Video Random Access Memory*.

ÍNDICE DE CONTENIDO COMPLETO

1. INTRODUCCIÓN

1.1. MOTIVACIÓN Y CONTEXTO

La enfermedad de Alzheimer es la forma más común de demencia, con más de 30 millones de casos en todo el mundo, número que tan solo se espera que se incremente en el futuro. La enfermedad viene caracterizada por un deterioro cognitivo progresivo e imparable, que incluye síntomas como pérdida de la memoria inmediata y a corto plazo, cambios conductuales y, finalmente, pérdida de funciones biológicas, que terminan produciendo la muerte.

Hay múltiples indicios de que patologías propias del alzhéimer pueden detectarse en pacientes años antes de la aparición de los síntomas propios de la enfermedad. Por ello existe un creciente interés en la búsqueda de biomarcadores útiles para el diagnóstico temprano de la enfermedad. Es de especial importancia encontrar biomarcadores para pacientes con deterioro cognitivo leve (*mild cognitive impairment*, MCI), de los que se estima que un 10% desarrollarán la enfermedad de Alzheimer en un periodo inferior a un año.

Algunos biomarcadores extraídos a partir de imágenes médicas e información genética se han utilizado para predecir con éxito el pronóstico en pacientes con desórdenes del comportamiento. La predicción conseguida ha resultado ser más precisa que la obtenida con otros instrumentos utilizados en la práctica clínica como escalas en test psicológicos o entrevistas estructuradas. Por ejemplo, se sabe que la corteza entorrinal y el hipocampo son las primeras estructuras afectadas por la pérdida de materia gris en el desarrollo de la enfermedad de Alzheimer, y que el alelo APOEe4 es el factor de riesgo genético más relevante. De entre las imágenes médicas utilizadas para la extracción de biomarcadores destacan las imágenes por resonancia magnética (*magnetic resonance imaging*, MRI), debido a su naturaleza no invasiva y coste menor a otras técnicas de imagen tridimensional. En la Figura 1.1. se muestra un ejemplo de este tipo de imágenes.

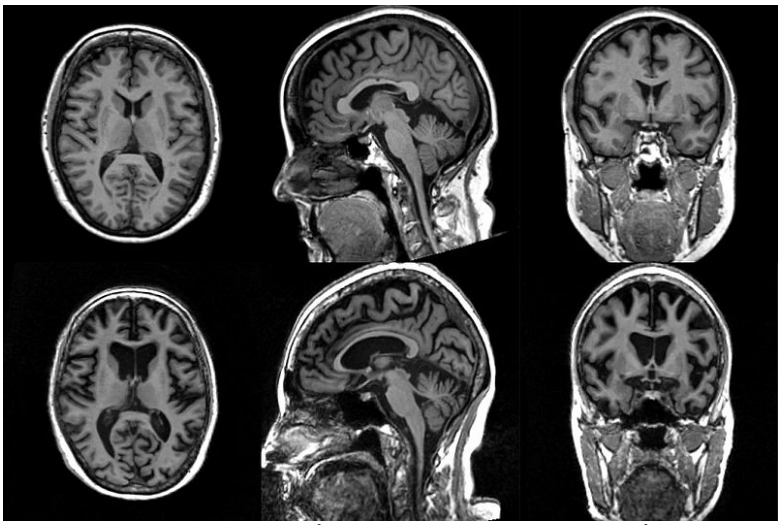


Figura 1.1 Ejemplos de imágenes por resonancia magnética (MRI). Vistas axial, sagital y coronal de dos imágenes 3D típicas de la base de datos de la Alzheimer's Disease Neuroimaging Initiative (ADNI).

La búsqueda de biomarcadores para la enfermedad de Alzheimer involucra la recopilación y tratamiento de una cantidad ingente de datos de orígenes muy diversos. Por este motivo, en los últimos años hay un creciente interés en el desarrollo de herramientas computacionales capaces de establecer qué información puede utilizarse en el diagnóstico y la detección temprana de la enfermedad.

La mayoría de los métodos desarrollados hacen uso de técnicas de aprendizaje supervisado y no supervisado y se pueden clasificar en tres grupos:

- En el primer grupo encontramos los modelos de regresión, técnica de aprendizaje supervisado utilizada para predecir la relación entre variables. Estos modelos intentan predecir la evolución de biomarcadores o de evaluaciones clínicas en función del tiempo, con el fin de caracterizar la evolución de estos datos en los distintos pacientes.
- En el segundo grupo encontramos las técnicas de aprendizaje automático. Con ellas se busca establecer relaciones entre los datos de los pacientes con etiquetas asignadas por un experto (e.g. enfermo/sano), para que el sistema aprenda a realizar el diagnóstico de la enfermedad. El método desarrollado en este TFG pertenece a esta categoría.
- En el tercer grupo encontramos técnicas de aprendizaje no supervisado, que intentan desarrollar modelos de la progresión de la enfermedad sin utilizar conocimientos clínicos previos. Estos métodos estudian la evolución concurrente de los biomarcadores de la enfermedad para intentar determinar en qué momento sus valores se convierten en anormales.

En la actualidad se han descubierto multitud de biomarcadores que podrían tener relación con la enfermedad:

- Tests cognitivos: tests neuropsicológicos que evalúan las capacidades cognitivas de los pacientes, como pueden ser la memoria, lenguaje, visión...
- Imágenes por resonancia magnética (MRI): técnica de adquisición de imágenes anatómicas. Las MRI cerebrales permiten visualizar la cantidad de materia blanca, gris y fluído cerebro-espinal del cerebro. La pérdida de volumen de las materias blanca y gris pueden ser un indicador de la enfermedad de Alzhéimer.
- Tomografía por emisión de positrones o PET: técnica de adquisición de imágenes que utiliza un radiofármaco introducido por vía venosa en el paciente que se enlaza a proteínas específicas. El PET aporta información sobre los procesos moleculares que tienen lugar en los órganos. Se sabe que en las fases tempranas del desarrollo de la enfermedad se pueden visualizar ciertas anormalidades en este tipo de imágenes.
- Proteínas en el líquido cefalorraquídeo: el líquido cefalorraquídeo es un líquido incoloro hallado en el cerebro y en la médula espinal. La concentración anormal de ciertas proteínas como la proteína Tau o la Beta-Amiloide en este líquido es uno de los primeros indicadores de la enfermedad.
- Factores de riesgo: entre estos se incluyen ciertos factores genéticos, clínicos, condiciones médicas, la edad de los pacientes y hábitos como fumar, sedentarismo y una dieta no saludable. El más importante es el alelo e4 del gen de la apolipoproteína E, o APOe4, el cual se estima puede aumentar las posibilidades de desarrollar alzhéimer entre 10 y 30 veces.

En la última década se han desarrollado una serie de iniciativas que organizan competiciones con el fin de avanzar en el conocimiento de la enfermedad de Alzheimer. La más reciente es TadPole challenge (https://tadpole.grand-challenge.org/), que pretende promover el desarrollo de herramientas computacionales predictivas en diferentes tareas relacionadas con el diagnóstico de la enfermedad. En concreto, TadPole se centra en la predicción del estado de un individuo, los valores esperados en distintos test cognitivos, y el volumen de los ventrículos cerebrales a partir de una serie de biomarcadores establecidos.

En este TFG vamos a abordar el problema de clasificación entre individuos sanos y enfermos (AD vs HC), junto con el de predicción de la conversión de MCI a enfermo en un plazo inferior a tres años (pMCI vs sMCI). En el sistema desarrollado utilizaremos los siguientes biomarcadores: datos clínicos (edad, sexo, resultados de test cognitivos…), información genética del APOEe4 e imágenes MRI.

1.2. Estado del arte

En los últimos años ha habido multitud de publicaciones donde se describen diferentes sistemas de clasificación para resolver el problema de distinguir entre individuos sanos y enfermos y el problema de predecir la evolución de los pacientes con MCI. La mayoría de estos sistemas utilizan técnicas de aprendizaje automático diversas sobre diferentes biomarcadores. Los marcadores más utilizados son imágenes de tipo PET y MRI, así como los resultados de diferentes tests cognitivos.

En estos métodos cabe destacar la diferenciación entre dos planteamientos para resolver el problema. En el primero se separa el proceso en dos pasos, uno inicial para reducir las dimensiones de los datos (la utilización de imágenes 3D resulta en sistemas con varios millones de valores de entrada) y un segundo paso para entrenar al clasificador dados los datos reducidos. El segundo planteamiento es el de usar *deep learning* sobre los datos sin reducir su dimensionalidad, y dejar que la red neuronal aprenda las reducciones necesarias de la dimensionalidad.

A continuación se comentan las técnicas y resultados obtenidos para la resolución del problema de pMCI vs sMCI que forman parte del estado del arte:

- En [2] se utilizan imágenes MRI junto a medidas cognitivas, LASSO para reducir la dimensionalidad de las imágenes MRI, y máquinas de soporte vectorial (SVM) para el clasificador. Con esto se obtiene una precisión del 82% y un AUC de 0.9.
- En [3] se utilizan los mismos datos de entrada que en la técnica anterior, *elastic net* para la reducción de dimensionalidad, y nuevamente SVM para el clasificador, obteniendo precisión de 84% y AUC de 0.92.
- En [4] se utilizan sólo imágenes MRI, se reduce la dimensionalidad aplicando técnicas de morfometría combinadas con un algoritmo genético basado en Test-t. Esto consigue valores de precisión del 75% y AUC de 0.75.
- Otros métodos utilizan más biomarcadores que los métodos anteriores. En [5] añaden PET, APOe4 y datos demográficos, y utilizan análisis de componentes independientes (ICA), para la reducción de la dimensionalidad y regresión de Cox para la clasificación. Obtienen un 85% de precisión y un 0.92 de AUC.
- Con datos clínicos, MRI, datos de proteómica en plasma y datos de medicaciones entrenan en [6] un modelo realizando la reducción con *joint mutual information* y la clasificación con *kernel learning*, obteniendo un 80% de

precisión y un 0.87 de AUC. Mediante el uso de rs-fMRI (*resting-state functional* MRI), métodos basados en grafos y SVM obtienen en [7] una precisión sobre el conjunto de validación del 91.4%, y un AUC de 0.95. Como estos resultados no son sobre un conjunto de test independiente no son tan significativos y no son comparables con el resto de resultados.

- Entre aquellos modelos que sólo utilizan datos PET encontramos dos aproximaciones de *deep learning*, una con redes neuronales convencionales [8] y otra con redes neuronales convolucionales [9]. Estos métodos obtienen, respectivamente, un 82.5% de precisión sin medida de AUC y un 84.2% de precisión con un 0.89 de AUC.
- En [1] se propone otra aproximación con redes neuronales convolucionales que, utilizando medidas cognitivas, APOe4, MRI y datos demográficos, obtiene una precisión del 86% y un AUC de 0.925. Este método obtiene los mejores resultados sobre un conjunto independiente de test hasta la fecha. Por este motivo, se ha elegido basar en este método el sistema de clasificación desarrollado en este TFG.

1.3. OBJETIVOS

Los objetivos que se pretende conseguir con el desarrollo de este proyecto son los siguientes :

- Implementar un sistema de clasificación para resolver los problemas de AD vs HC y pMCI vs sMCI, basado en la red neuronal descrita en [1]. Dicha implementación se realizará en Python 3.6, con el uso de las bibliotecas Tensorflow y Keras, así como de parte del código proporcionada por los autores de [1].
- Generar diferentes modelos entrenando la red neuronal con diferentes combinaciones de los datos de entrada utilizados en [1] y diferentes normalizaciones de las imágenes MRI.
- Reproducir los resultados obtenidos en [1] para los experimentos normalizados respecto al atlas MNI152 mediante ANTS-SSD.
- Identificar qué combinación de los datos de entrada y qué método de normalización obtienen los mejores resultados en los problemas de clasificación.
- Utilizar diversas técnicas para la interpretación de redes neuronales con el fin de comprender los resultados e identificar las posibles limitaciones del sistema.
- Proponer alternativas para la mejora del sistema.

1.4. ESTRUCTURA DE LA MEMORIA

En el capítulo 1 se introduce el contexto de este TFG, se describe el estado del arte y se enumeran los objetivos que se pretenden conseguir.

En el capítulo 2 se presenta, en primer lugar, una descripción detallada de los conceptos de *deep learning* y CNNs utilizados para la elaboración de este TFG. A continuación se describe la arquitectura de la red convolucional utilizada, así como del sistema de clasificación. Por último se detallan los datos utilizados, el proceso de entrenamiento de la red neuronal y se termina con una descripción del equipo utilizado para ejecutar los experimentos así como algunos detalles técnicos de la implementación.

En el capítulo 3 se presenta una selección de los experimentos realizados y se analizan los resultados obtenidos.

Por último, en el capítulo 4, se desarrollan las conclusiones más importantes del proyecto y se proponen las líneas futuras de trabajo.

2. MÉTODO DESARROLLADO

A lo largo de este capítulo se presentan los fundamentos de *deep learning* y redes neuronales convolucionales necesarios para la comprensión de la red implementada en este TFG. A continuación se incluye una descripción detallada de todas las capas utilizadas en la misma, se detalla la arquitectura de la red y el proceso completo para realizar la clasificación y predicción partiendo de los datos de entrada. Por último se presenta el conjunto de datos sobre el que se han realizado los experimentos, se detalla el proceso que se ha seguido para entrenar la red y se especifican los parámetros utilizados.

2.1. DEEP LEARNING

Un algoritmo de *deep learning* es un método de aprendizaje automático que toma como entrada un dato *x* y lo usa para predecir un valor *y*. Dado un conjunto de datos de entrada y valores de salida, el algoritmo intentará aproximar las predicciones que produce a los valores de salida esperados. Para realizar dichas predicciones se utilizan unas estructuras conocidas como redes neuronales. La palabra *deep* hace referencia al número de capas que conforman la red, que es bastante superior al utilizado en redes neuronales convencionales.

2.1.1. Redes neuronales

Una red neuronal está compuesta por capas conectadas entre sí de forma secuencial, las cuales a su vez están compuestas de neuronas. A la primera capa se la denomina capa de entrada, pues es la que recibe los datos x de entrada del algoritmo, a la última capa se la denomina capa de salida, pues es la que calcula las predicciones, y a todas las capas intermedias se las denomina capas ocultas. El número de capas ocultas o profundidad de la red a menudo determina la complejidad y el rendimiento de la misma. En la figura 2.1 puede verse un ejemplo de red neuronal con dos capas ocultas.

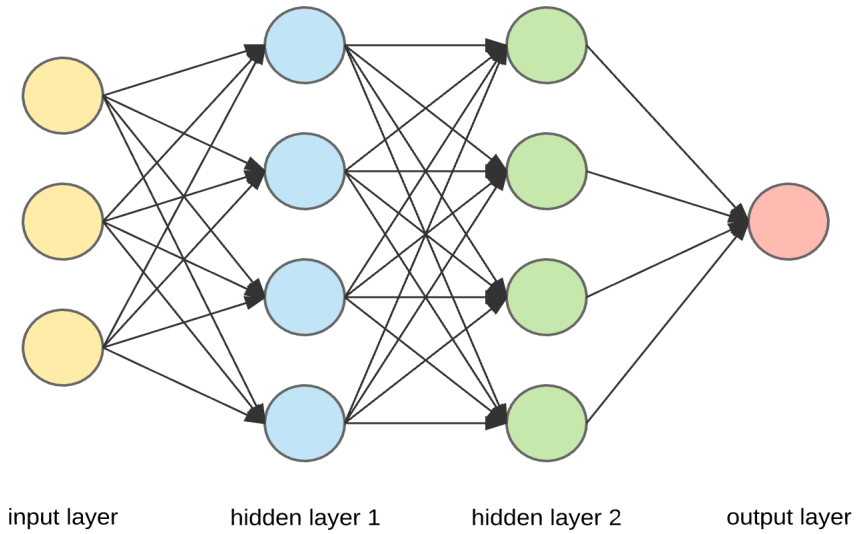


Figura 2.1 Ejemplo de una red neuronal con cuatro capas, dos de ellas ocultas.

Cada neurona tiene como entrada un vector de *n* valores reales, x, y devuelve un único valor real como salida. Para ello cada neurona viene caracterizada por dos parámetros, un vector de pesos *w*, de tamaño igual a su número de entradas, y un valor denominado bias *b*. Adicionalmente, requiere de una función *f* no lineal denominada función de activación. La salida de una neurona viene dada por la siguiente ecuación:

$$f(\sum_{n} w_n x_n + b)$$

En la figura 2.2 puede verse una representación visual de los elementos de una neurona.

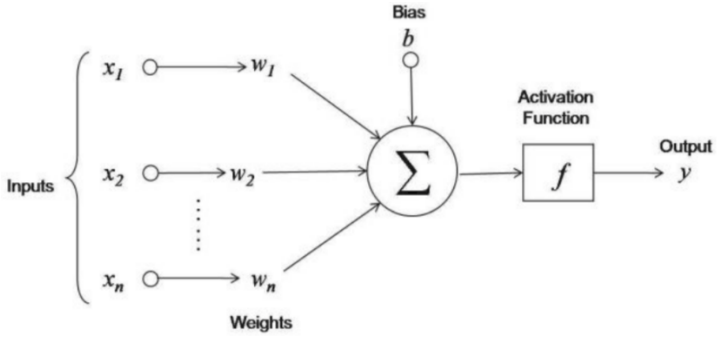


Figura 2.2 Representación visual de una neurona. Imagen obtenida de https://towardsdatascience.com/statistics-is-freaking-hard-wtf-is-activation-function-df8342cdf292

La función de activación f es fundamental, pues sin un elemento que añada no linealidad al cálculo, las redes neuronales de varias capas no podrían aprender funciones más complejas que las redes de una sola capa. Hay multitud de funciones de activación, como sigmoidal, tanh, ReLU,ELU, etc. En la figura 2.3 pueden verse las gráficas y ecuaciones de algunas de dichas funciones.

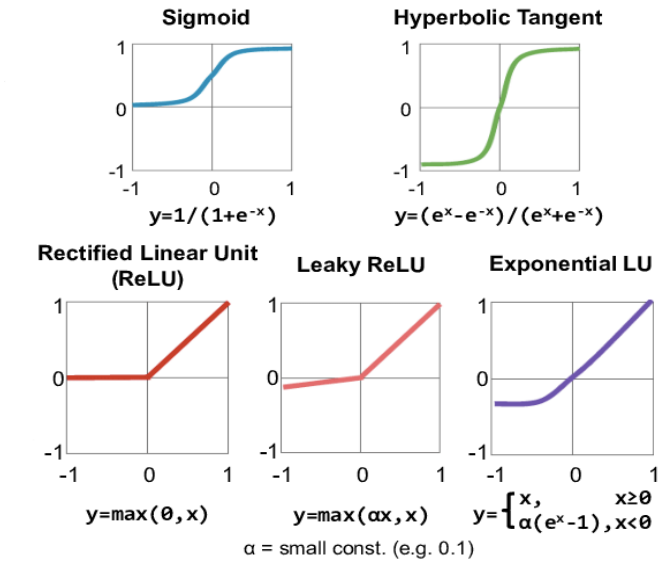


Figura 2.3 Gráficas de algunas de las funciones de activación más comunes. Imagen obtenida de [10].

Para que una red neuronal realice predicciones acertadas hay que seleccionar los valores apropiados para los pesos y bias de cada una de sus capas. Esto se hace mediante un proceso denominado entrenamiento.

2.1.2. Entrenamiento de una red neuronal

El primer paso para describir el proceso de entrenamiento de una red neuronal consiste en definir una métrica que cuantifique cómo de buenas son las predicciones que esta es capaz de realizar. Dicha métrica viene asociada a una función de coste o error. La más utilizada se denomina *binary cross entropy* y se define de la siguiente manera, siendo y el vector de predicciones realizadas por la red, $\hat{y}$ el vector de los valores correctos esperados y N el número total de predicciones:

$$L = -\frac{1}{N}\sum_{i=1}^{N} y_i \cdot log(\hat{y}_i) + (1-y_i) \cdot log(1-\hat{y}_i)$$

El objetivo del entrenamiento de una red neuronal es el de minimizar la función de coste, mediante el ajuste de los parámetros de la red (pesos y *bias* de cada neurona). Esto se realiza mediante un algoritmo de optimización que suele basarse en un gradiente descendente estocástico. Para ello, es necesario el cálculo del gradiente de la función de coste que se realiza mediante el método de *backpropagation*.

El algoritmo de *backpropagation* consiste en el cálculo de la derivada de la función de coste con respecto a cada peso de la red utilizando técnicas de diferenciación algorítmica. Para calcular la derivada de los pesos de cada capa es necesaria la derivada de la capa siguiente, de acuerdo con la regla de la cadena. Así, el cálculo de las derivadas se realiza desde la última capa hasta la primera.

Una vez calculado el gradiente, se han de actualizar los pesos de la red de acuerdo al algoritmo de optimización utilizado. Para el método de gradiente descendente estocástico, dado W_n los pesos de una capa, W_{n+1} los pesos tras ser actualizados, y α el factor de aprendizaje, la actualización se realiza mediante la fórmula:

$$W_{n+1} = W_n - \alpha \frac{\partial L}{\partial W_n}$$

El método de optimización es estocástico porque en cada actualización el gradiente se calcula a partir de una serie de muestras elegidas de manera aleatoria, denominadas *batch*. El proceso completo de entrenamiento de una red neuronal sigue los siguientes pasos:

- Dado un conjunto de datos usados para el entrenamiento, calcular sus predicciones.
- Calcular la función de coste con las predicciones del paso anterior y los valores esperados.
- Calcular el gradiente de la función de coste mediante backpropagation.
- Actualizar todos los pesos de la red.

Este proceso se realiza iterativamente de manera que en cada iteración se reduce el valor de la función de coste. Así, cuantas más iteraciones realicemos menor será la función de coste hasta que se alcancen las condiciones de convergencia.

No obstante, en ocasiones, la minimización de la función de coste no implica que el modelo entrenado sea mejor. Es posible que el modelo se ajuste demasiado a los datos con los que está entrenando, y pierda capacidad predictiva con datos que no ha visto nunca. A este fenómeno se le denomina *overfitting*. Notar que este fenómeno no es exclusivo de las redes neuronales.

Si queremos evaluar el poder predictivo de una red neuronal entrenada, hay que tener en cuenta una serie de consideraciones. Como el objetivo del sistema es el de realizar predicciones correctas sobre datos nuevos, es común reservar un conjunto de datos para la supervisión del proceso de entrenamiento. Éstos se denominan datos de validación, y son utilizados en ocasiones para ajustar los hiper-parámetros del modelo. Además se suele reservar otro conjunto de datos, denominados de test, para una evaluación final del sistema. Es importante recalcar que nunca han de usarse los datos de validación ni los de test para entrenar los pesos de la red.

Para evaluar si un sistema está cometiendo *overfitting*, una técnica común es la de calcular, para cada iteración, la función de coste dadas las predicciones del conjunto de datos de validación para compararla con el coste sobre conjunto de entrenamiento. En la figura 2.4 se muestra un ejemplo típico de las curvas de los errores de entrenamiento y de validación durante el entrenamiento de un sistema de aprendizaje que termina cometiendo *overfitting*. Al inicio del entrenamiento, tanto el error de validación como el de entrenamiento son altos y decrecen a un ritmo similar. Cuando comienza el *overfitting*, el error de validación empieza a aumentar mientras que el de entrenamiento sigue decreciendo. Esto significa que el modelo se ha especializado demasiado en predecir los datos de entrenamiento y ha perdido la capacidad de predecir datos no vistos previamente.

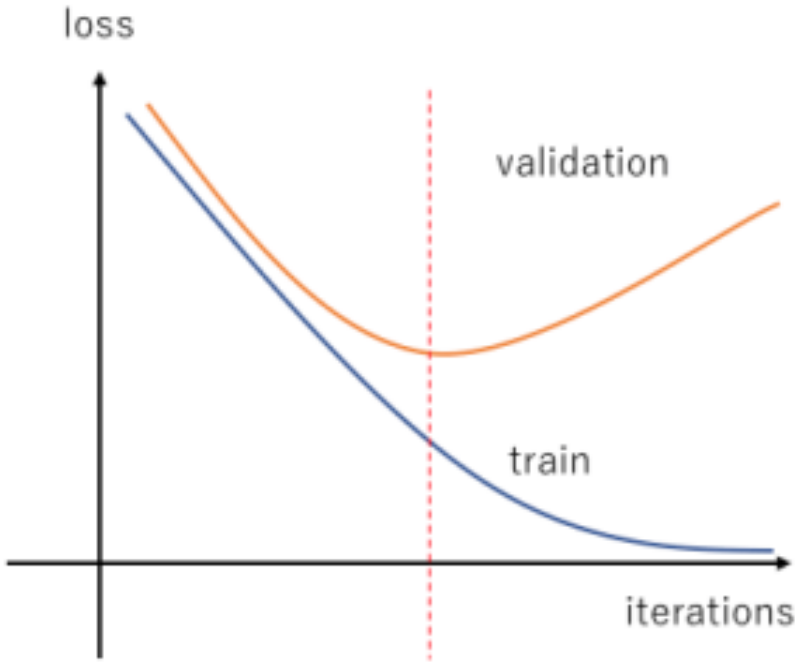


Figura 2.4 Curvas de los costes de validación (naranja) y entrenamiento (azul) en un sistema con overfitting. Imagen obtenida de http://mlexplained.com/2018/04/24/overfitting-isnt-simple-overfitting-re-explained-with-priors-biases-and-no-free-lunch/

Dado que el problema de *overfitting* afecta a la mayoría de algoritmos de aprendizaje automático, existen multitud de estrategias para combatirlo que son aplicables al entrenamiento de redes neuronales. Dichas estrategias reciben el nombre de regularización. Las más utilizadas son las siguientes:

- Aumentar el número de datos de entrenamiento: tener pocos datos de entrenamiento es una de las principales causas de *overfitting*. El simple hecho de entrenar sobre más datos dificulta al modelo el sobre ajustarse a unos pocos pues necesita ser lo bastante general para predecir con el menor error posible todos los datos de entrenamiento.

- *Early stopping*: consiste en parar el entrenamiento antes de que se empiece a cometer *overfitting*. Para ello se suele escoger el número de iteraciones de entrenamiento que minimice el error de validación.

- *Dropout*: esta técnica sencilla pero comúnmente utilizada, consiste en cancelar la salida de algunas neuronas de cada capa durante el entrenamiento con una probabilidad p, normalmente p = 0.5. La idea intuitiva de por qué este proceso ayuda a regularizar el modelo es que evita que la red neuronal dé demasiada importancia a ciertos pesos de la red, pues en cualquier iteración este valor puede no estar disponible para realizar la predicción [11].

- Regularización L2 de la matriz de pesos: tiene como objetivo penalizar al modelo por utilizar pesos grandes. Para ello suma un término a la función de coste, que consiste en la suma del cuadrado de todos los pesos, multiplicado por un parámetro. La selección de dicho parámetro es crucial para la efectividad de la regularización, y suele necesitar de una búsqueda exhaustiva o ser estimado como hiper-parámetro del sistema.

2.2. Redes neuronales convolucionales

Una red neuronal convolucional (*convolutional neural network*, CNN) es un algoritmo de *deep learning* basado en una arquitectura de red similar en muchos aspectos a la de una red neuronal convencional, pero diseñado para trabajar con datos de dos o más dimensiones, generalmente imágenes. En la última década se ha producido una explosión de la popularidad de los algoritmos de *deep learning* gracias a la superioridad de las CNNs en una infinidad de tareas de visión artificial. En la actualidad, la investigación con CNNs se ha extendido a disciplinas muy diferentes.

2.2.1. Convolución

La operación principal de las CNNs es la convolución, la cual está definida sobre un volumen de entrada (la imagen), una matriz de valores con el mismo número de dimensiones que el volumen, pero normalmente de tamaño mucho menor, denominada *kernel* o filtro, y un valor para el desplazamiento del *kernel* conocido como paso o *stride*.

El resultado de la convolución es otro volumen, con el mismo número de dimensiones que el volumen original, donde cada valor es el resultado de multiplicar el *kernel* con una porción de el volumen de entrada. El *kernel* se desplaza sobre dicho volumen de la siguiente manera, inicialmente se coloca en una esquina, y avanza sobre una dimensión un número de posiciones igual al paso. Cuando llega al final de dicha dimensión, vuelve al comienzo de esta y avanza una vez sobre la siguiente dimensión, tras lo cual se repite el proceso. Esto es generalizable para cualquier número de dimensiones del volumen de entrada.

En la figura 2.5 puede verse una visualización de la operación de convolución para un volumen de entrada de dimensiones 5x5 y un *kernel* 3x3:

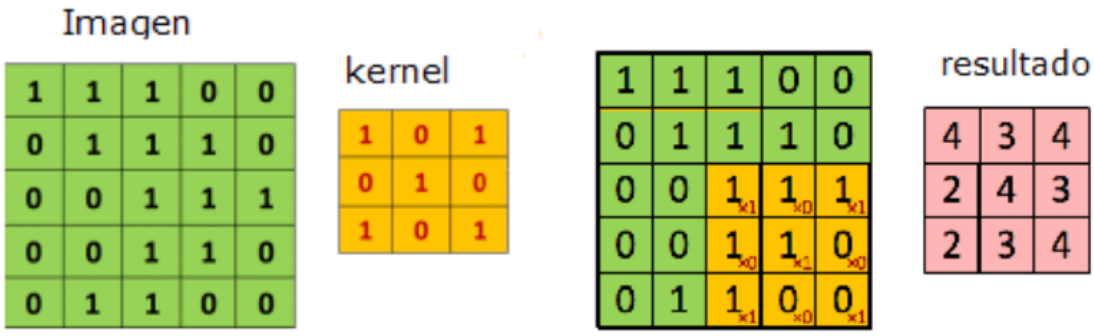


Figura 2.5 Ejemplo del efecto de una convolución 2d al aplicar un kernel(amarillo) a una imagen(verde), obteniendo otra imagen como resultado(rojo). Imagen obtenida de http://deeplearning.stanford.edu/wiki/index.php/Feature_extraction_using_convolution

En general, el tamaño del volumen de salida depende del tamaño de la entrada, del tamaño del *kernel*, del paso y de si se ha aplicado *padding* sobre el volumen. El *padding* consiste en aumentar las dimensiones del volumen de entrada, normalmente añadiendo ceros en los márgenes, para mantener el tamaño del mismo tras aplicar la convolución. En el caso general, para cada dimensión del volumen, si el tamaño original es n, el tamaño del *kernel* en esa dimensión es k, el paso es s y se ha añadido *padding* de tamaño p, el tamaño del resultado d será:

$$d = \frac{n+2p-k}{s} + 1$$

El uso de convoluciones es una técnica común en análisis de imagen y visión por computador. Por ejemplo, la detección de los contornos de una imagen puede realizarse mediante convoluciones. El filtro de Sobel-Fedman consiste en realizar n convoluciones con n *kernels* diferentes sobre una imagen de n dimensiones, para calcular una aproximación del gradiente de la misma. Las zonas de gradientes más grandes representan los contornos de la imagen. A continuación se muestra uno de los *kernels* del operador de Sobel, que calcula el gradiente en la dimensión X de la imagen.

$$\begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$$

La figura 2.6 muestra un ejemplo del resultado de la convolución de este *kernel* aplicado a una imagen 2D.

Figura 2.6 Ejemplo de convolución con el kernel de Sobel aplicado a una imagen. Izquierda, imagen original. Derecha, resultado de la convolución (gradiente horizontal).

2.2.2. Capas convolucionales

Las convoluciones dentro de una CNN se utilizan agrupadas en capas. Cada capa está compuesta por un número K^l de filtros, y cada uno de estos se aplica sobre la entrada de dicha capa. El número de filtros, así como el tamaño de los mismos, y el *stride* que se aplica a las convoluciones son parámetros de la arquitectura de la CNN. Por el contrario, los valores reales del *kernel* serán las variables de aprendizaje de la red, así como un valor adicional, conocido como bias, que se suma al resultado obtenido de cada multiplicación del *kernel* con parte de la entrada. Al resultado de cada filtro se le denomina canal o activación, y el resultado de cada capa se obtiene de concatenar sobre una nueva dimensión cada canal. Los datos de entrada de la red también pueden estar separados en canales, por ejemplo las imágenes a color suelen separarse en 3 canales, uno para cada color (RGB). Debido a esto, cada filtro debe de tener una dimensión extra cuyo tamaño debe coincidir con el número de canales de su entrada, ya sea la imagen original o la capa anterior. Tras obtener los resultados de las convoluciones, al igual que con las redes neuronales tradicionales es importante introducir un elemento de no-linealidad, en la forma de una función de activación, que se aplica sobre cada valor del volumen. En la figura 2.7 puede observarse cómo se aplica sobre una entrada con tres canales una convolución con un filtro 3x3x3:

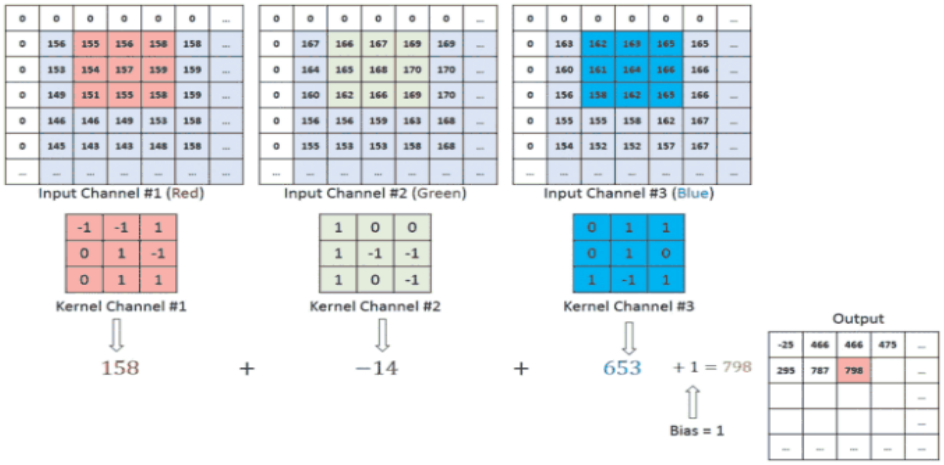


Figura 2.7 Ejemplo de convolución con 3 canales. Los tres canales de la imagen pueden observarse en la parte superior, seguidos de un kernel tridimensional. En la esquina inferior derecha se observa el resultado de la convolución. Imagen obtenida de ttp://machinelearninguru.com/computer_vision/basics/convolution/convolution_layer.html .

2.2.3. Convoluciones 3D

Al referirse a la dimensionalidad de las convoluciones utilizadas en una CNN, se suele omitir la última dimensión correspondiente al número de canales. Por tanto, aunque los filtros utilizados en las explicaciones del apartado anterior tengan tres dimensiones, se les denominaría 2D porque como el tamaño del filtro en la última dimensión coincide con el de la entrada, siendo este el número de canales, el kernel no se desplaza al realizar la convolución sobre esta dimensión. Las convoluciones 3D son aquellas que se aplican sobre volúmenes de cuatro dimensiones, incluyendo la dimensión de los canales, y que usan filtros de cuatro dimensiones.

Es importante considerar el número de variables de aprendizaje presentes en cualquier modelo de aprendizaje automático, pues esto supone un factor restrictor en el diseño. A mayor número de parámetros más complejo es el modelo, y en principio puede predecir mejor, pero también es más propenso a sufrir de *overfitting*. Si llamamos a las dimensiones del kernel de una capa N^1 , N^2 y N^3 , al número de canales de la entrada K^{l-1} y al número de filtros K^l , el número total de parámetros de la capa viene dado por :

$$\left(N^1 * N^2 * N^3 * K^{l-1} + 1\right) * K^l$$

2.2.4. Convoluciones separables

Una convolución separable es una operación similar a las empleadas en las capas convolucionales convencionales. Esta operación se compone de dos pasos, primero se aplican convoluciones por separado a cada uno de los canales de entrada, y después estos se juntan mediante una convolución con un kernel puntual para obtener el resultado final. Los kernels puntuales tienen tamaño 1 en todas dimensiones menos la última, que coincide con el número de canales de la entrada, y sirven para extraer las correlaciones entre canales.

La justificación tras esta operación reside en la hipótesis de que las correlaciones espaciales en los volúmenes y las correlaciones entre canales pueden tratarse por separado en vez de conjuntamente con una única convolución [12].

En detalle, cada capa separable contará con K^{l-1} filtros, uno por cada canal de la entrada, cuyo número de dimensiones debe coincidir con el de cada canal individual, es decir, tendrán una dimensión menos que los filtros convolucionales normales. Adicionalmente contará de K^l filtros puntuales, y por tanto generará un volumen resultado con K^l canales .

La convolución separable se desarrolla de la siguiente manera. En primer lugar, sobre cada canal de entrada se aplica una convolución con su filtro correspondiente, obteniendo así K^{l-1} volúmenes intermedios. Después, se aplica una nueva convolución para cada filtro puntual sobre todos los volúmenes intermedios, obteniendo para cada uno un canal del volumen resultado. En la figura 2.8 se puede observar un esquema de este proceso.

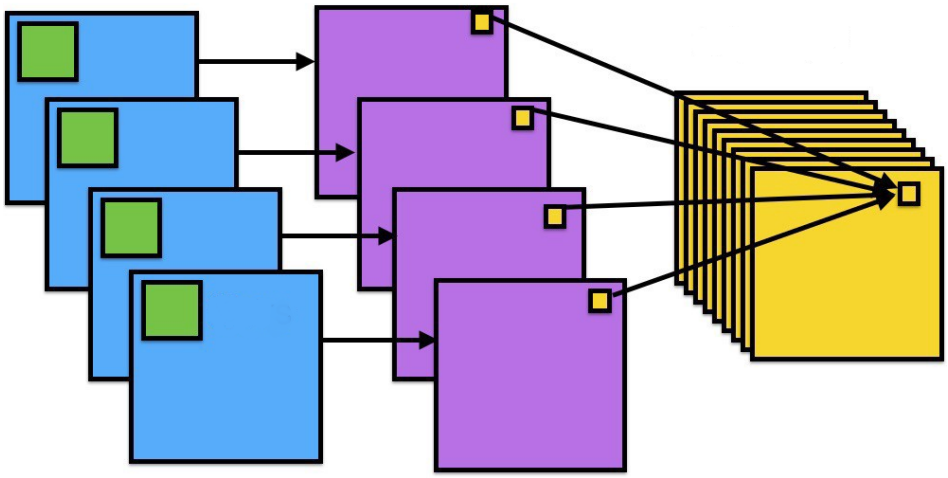


Figura 2.8 Convolución separable. Pueden observarse los canales originales de entrada (azul),junto los filtros aplicados a cada uno de ello (verde). Seguidamente aparecen los volúmenes intermedios (morado) y las convoluciones puntuales (amarillo, en el centro) de dimensión 1x1x4. Por último aparecen los distintos canales del volumen resultado(amarillo, derecha). Imagen obtenida de https://www.deeplearningitalia.com/guia-para-arquitecturas-de-redes-profundas/

Utilizando la misma nomenclatura que para las convoluciones 3D, se puede calcular fácilmente el número de parámetros de una capa con convoluciones separables, que vendrá dado por:

$$\left(N^1 * N^2 * N^3\right) * K^{l-1} + (K^l + 1) * K^{l-1}$$

Si tomamos como ejemplo una de las capas utilizadas en la red neuronal, explicada en detalle en la sección 2.2, podemos comparar la reducción en el número de parámetros conseguida por utilizar convoluciones separables. Así, para unas dimensiones del filtro de 3x4x3, 96 canales de entrada y 96 filtros en la capa obtenemos que:

- Convoluciones convencionales: 331.968 parámetros.
- Convoluciones separables: 12.768 parámetros.

Es decir, sustituir una capa por una con convoluciones separables supone un decremento de alrededor 26 veces el número de parámetros utilizados en dicha capa. Como ventaja adicional, las convoluciones separables realizan menos operaciones que las convencionales, luego son más rápidas de evaluar.

2.2.5. Pooling

Otra operación introducida en las CNN es el *pooling*. Existen dos tipos de *pooling*, *average pooling* y *max pooling*. La operación de *pooling* es similar a una convolución, consiste en desplazar un kernel sobre el volumen, de igual manera que para las convoluciones, pero esta vez el resultado será el valor mayor del volumen de los cubiertos por el kernel en el caso del *max pooling*, o la media de los mismos en el caso del *average pooling*. En la figura 2.9 se puede observar el resultado de aplicar un *max pooling* de dimensión 3x3, sobre una imagen de dimensión 5x5.

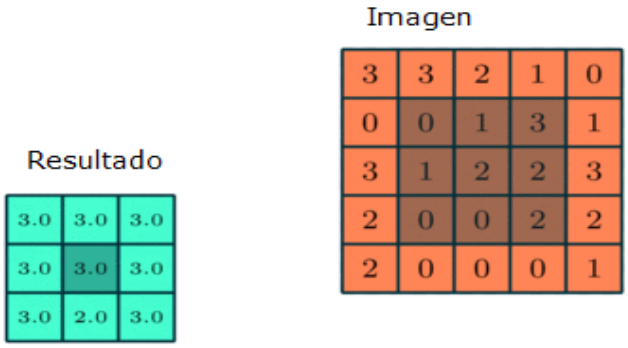


Figura 2.9 Operación de Max pooling sobre una imagen(naranja), obteniendo otra imagen resultado(azul). Imagen obtenida de http://deeplearning.stanford.edu/wiki/index.php/Feature_extraction_using_convolution.

Un uso habitual del *pooling* es el de disminuir el tamaño de los volúmenes, para así reducir el coste computacional de aplicar convoluciones sobre los mismos. Adicionalmente, el *max pooling* es útil para destacar las activaciones máximas de la red, lo que puede facilitar la retención de información de ciertos detalles.

2.2.6. Capas completamente conectadas

Las capas completamente conectadas (*fully connected*, FC), son el bloque principal de las redes neuronales tradicionales, aunque también son utilizadas comúnmente en redes neuronales convolucionales, en especial en las últimas capas y para realizar las predicciones finales. La entrada de una capa FC tiene que ser un vector de una sola dimensión, de manera que para conectarla con la salida de una capa convolucional el volumen se tiene que aplanar.

2.2.7. Batch normalization

Batch normalization es una operación que se aplica sobre los volúmenes resultado de la capa anterior. Por cada activación de la capa anterior, *batch normalization* tiene dos parámetros entrenables, γ y β. Tiene como peculiaridad que trata a todos los volúmenes correspondientes a cada muestra de un *batch* conjuntamente.

En primer lugar, requiere del cálculo de la media y la varianza de cada activación de un filtro de la capa anterior, a lo largo de todo el *batch* de entrenamiento. Tras esto estandariza dichas activaciones, es decir, les resta su media y las divide por su desviación estándar. Esto resulta en que, suponiendo que todas las muestras del *batch* provengan de la misma distribución, la media y varianza de cada activación sean aproximadamente 0 y 1 respectivamente. Tras esto, dado $\hat{x}$ cada activación normalizada, realiza la siguiente operación:

$$y = \gamma\hat{x} + \beta$$

Añadir *batch normalization* tras cada capa convolucional tiene como beneficio reducir el tiempo de entrenamiento requerido, pues permite el uso de un factor de aprendizaje mayor. También tiene un efecto regularizador, al tratar en conjunto cada muestra junto con las demás pertenecientes al mismo *batch*[13].

2.3. RED NEURONAL DESARROLLADA

La red neuronal que se ha desarrollado en este TFG, presenta tres entradas de datos diferenciadas. La primera es para datos vectorizados que se usará para dar entrada a los datos clínicos. La segunda es para matrices 3D, que se usará para las imágenes MRI a las que se realizará un preprocesado habitual en los estudios de Anatomía Computacional seguido de un alineamiento no-rígido a un sistema de coordenadas común, que en este trabajo será el del atlas MNI152 (normalización). La tercera también es para matrices 3D, y se usará para incluir información extraída del alineamiento no-rígido de las imágenes. En este trabajo se considerará el determinante de los Jacobianos de las transformaciones no-rígidas aplicadas en el proceso de normalización. Las tres entradas de datos son opcionales, de manera que la red puede alimentarse con cualquier combinación de estas.

El entrenamiento de la red se realizará sobre dos tareas distintas simultáneamente, a lo que nos referiremos como *joint learning*. Además, ambas tareas compartirán gran parte de las capas de la red. La primera tarea es la clasificación entre pacientes sanos, del grupo de control, y pacientes con la enfermedad de Alzheimer. La segunda tarea es predictiva, en esta se clasificarán pacientes de MCI en dos grupos: aquellos que van a desarrollar alzhéimer en un plazo de tres años y aquellos que no.

El objetivo del *joint learning* es que las capas compartidas aprendan a extraer características útiles a ambos problemas. Asumimos por tanto que existe cierta similitud entre ambas tareas, en concreto podrían considerarse los pacientes de MCI como un espectro entre los dos extremos formados por pacientes sanos y pacientes con enfermedad de Alzheimer. Por esto se espera que transformaciones similares a los datos de entrada proporcionen características útiles a ambos problemas de clasificación.

El proceso de *joint learning* presenta muchas ventajas respecto al entrenamiento por separado de las tareas. En primer lugar incrementa el número de muestras con las que podemos entrenar la red, lo cual es importante considerando la limitada cantidad de datos de la que disponemos. Adicionalmente, el tener que satisfacer ambas tareas puede verse como una restricción sobre los parámetros de la red y, por tanto, tener un efecto regularizador que ayude a reducir el *overfitting*.

La red está compuesta por una subred, a la que llamaremos extractor de características, que contiene todas las capas comunes a las dos tareas de clasificación, AD vs HC y pMCI vs sMCI. Dicha subred tiene como salida un vector de 4 componentes, valores a los cuales denominaremos características. Para cada problema de clasificación, una última capa completamente conectada tomará como entrada las características y realizará la predicción final. Un esquema de alto nivel de la red neuronal puede verse en la figura 2.10.

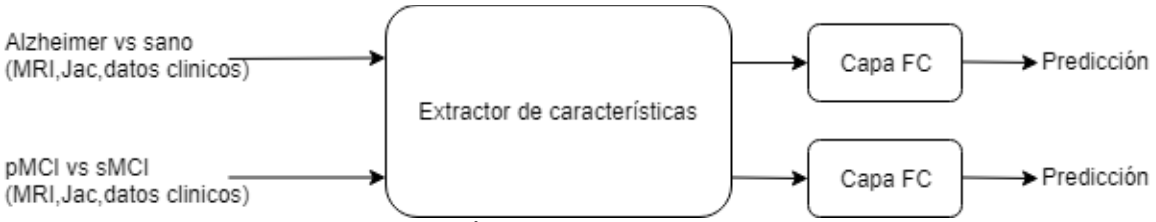


Figura 2.10 Esquema de la implementación del aprendizaje multitarea. Los datos de ambos problemas atraviesan el extractor, tras lo cual las características son procesadas por dos capas FC diferentes.

2.3.1. Bloques

Es habitual dividir la estructura de una CNN en bloques de capas que se utilicen múltiples veces dentro de la misma. En la CNN utilizada identificamos tres bloques distintos, uno para las convoluciones 3D, otro para las convoluciones separables y el último para las capas completamente conectadas (FC). Un esquema de los bloques puede verse en la figura 2.11. Los parámetros vienen indicados entre paréntesis, algunos con valores por defecto que se utilizarán a no ser que se indique lo contrario.

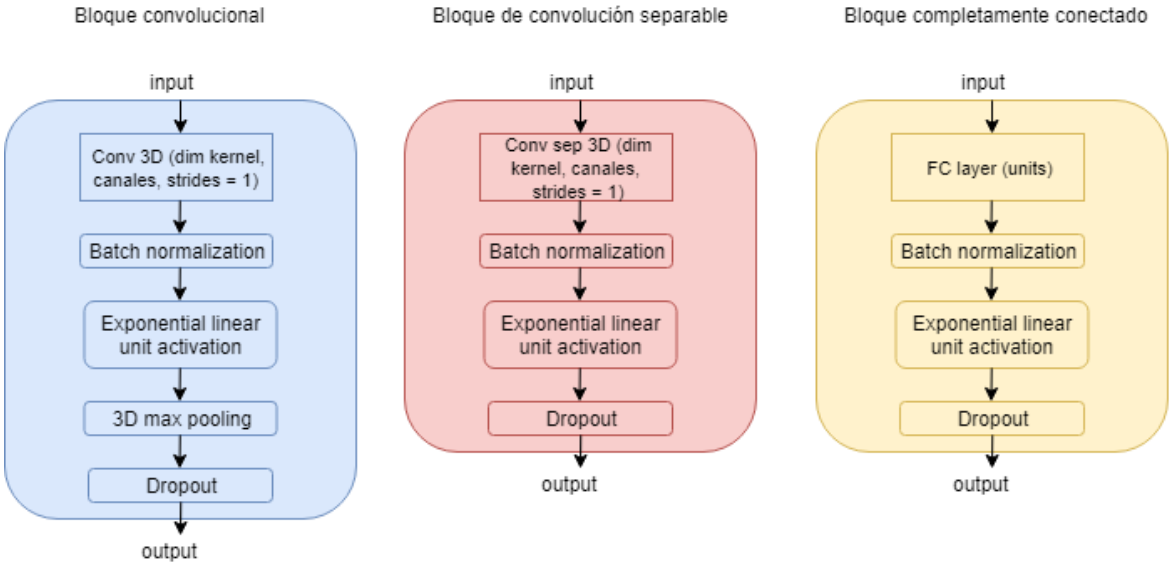


Figura 2.11 Muestra un esquema de los bloques convolucionales(azul), convolucionales separables(rojo) y completamente conectados(amarillo).

El bloque convolucional comienza con una convolución 3D, seguido de *batch normalization* y una función de activación ELU. Tras esto se aplica un 3D *max pooling*, para reducir las dimensiones de los volúmenes para reducir el coste computacional, y por último se aplica *dropout*. El bloque con convoluciones separables es muy similar al anterior, la diferencia es que no contiene una capa de max pooling. Esto es así para no reducir demasiado el tamaño de los volúmenes. Los parámetros de ambos bloques son las dimensiones de los filtros utilizados, el número de canales de salida, es decir el número de filtros a utilizar, y el paso de la convolución.

El tercer y último bloque utilizado es el completamente conectado, que comienza con una capa FC y continua nuevamente con *batch normalization*, ELU y *dropout*. Su único parámetro es el número de neuronas de la capa FC.

Nótese que la entrada de los bloques convolucionales ha de ser volúmenes 4D mientras que para el bloque completamente conectado ha de ser un vector con una única dimensión.

2.3.2. Arquitectura

En la figura 2.12 puede verse un esquema detallado de la arquitectura de la red. El tamaño de los datos de entrada para los dos tipos de volúmenes tridimensionales es de 182x218x182x1, con un único canal pues son imágenes en escala de grises. El de los datos clínicos es un vector con 13 componentes. En la parte izquierda de dicha figura puede verse el tamaño de los canales de salida tras cada bloque.

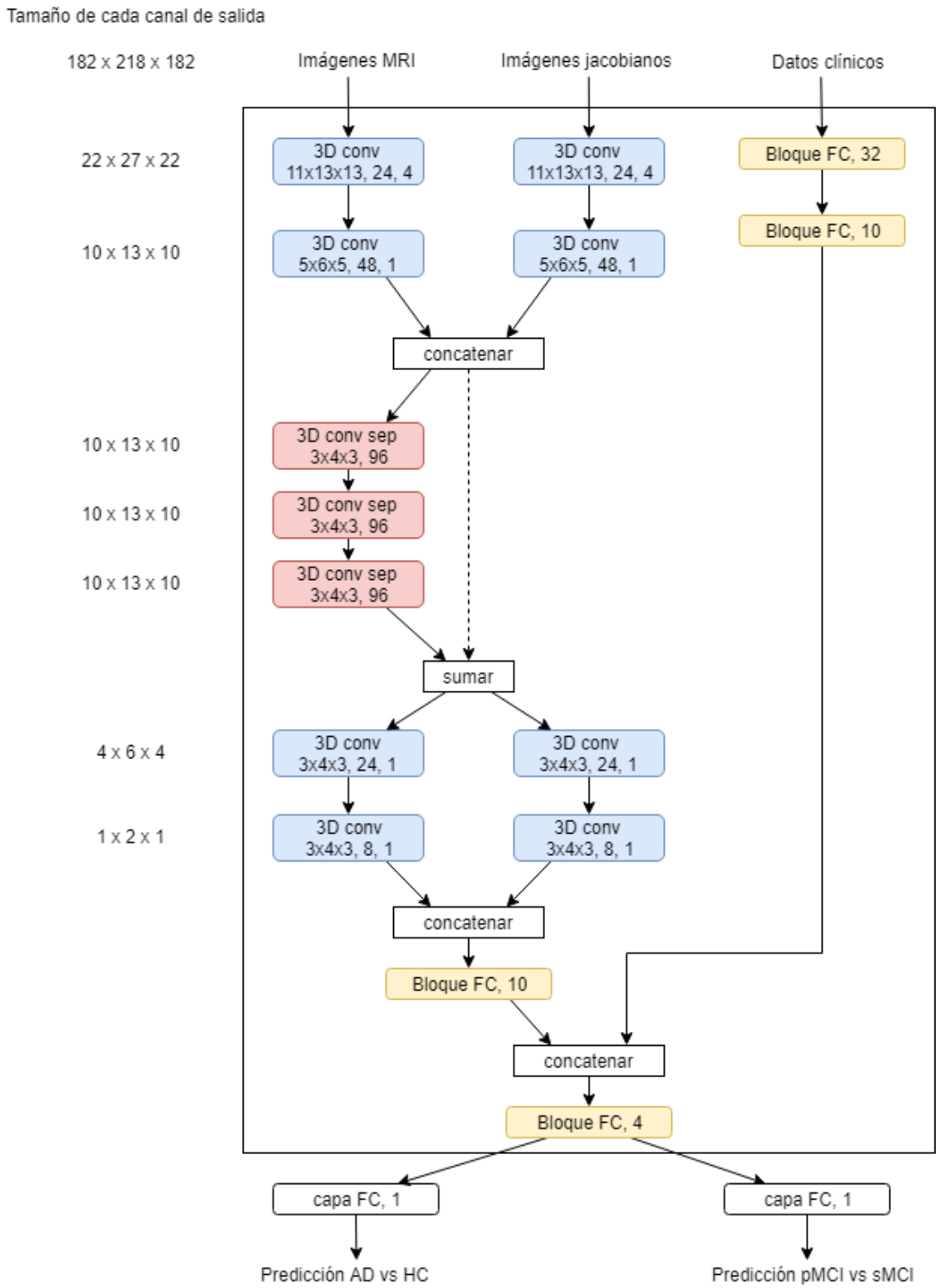


Figura 2.12 Esquema de la arquitectura de la red neuronal. Las entradas de datos aparecen en la parte superior, y las flechas simbolizan la ruta que toman dichos datos. En el margen izquierdo aparecen las dimensiones de los canales de salida de cada capa.

En las primeras capas las tres entradas se tratan por separado. Las imágenes tridimensionales de los MRI y los jacobianos pasan primero cada una por dos bloques convolucionales. Nótese que en el primero se utiliza un valor para el *stride* muy alto, de cuatro, y un número relativamente bajo de filtros, 24. Esto es así debido al gran coste computacional de las convoluciones 3D, y al usar un *stride* grande no solo reducimos considerablemente el tiempo de ejecución de dicha convolución, aproximadamente 4 veces frente a si usáramos un stride de 1, sino que además reducimos en mayor medida el tamaño de los canales resultado, acelerando todo el resto de la red neuronal.

Tras estos dos bloques, ambos volúmenes se unen con una operación de concatenación a través de la dimensión de los canales, es decir, que los canales del volumen resultante mantendrán la misma dimensión pero el número de canales de dicho volumen será la suma de los canales resultantes de las convoluciones aplicadas a los MRI y a los jacobianos. Después de dos bloques convolucionales ya se han reducido suficiente las dimensiones de los canales y aplicamos tres bloques de convoluciones separables, con *stride* de 1 y ajustando el *padding* de tal manera que el tamaño de los canales de salida se mantenga. Al usar convoluciones separables que tienen menos parámetros y son más rápidas que las convencionales, podemos permitirnos utilizar un número más alto de filtros, en concreto 96, con la esperanza de que la mayoría del proceso de extracción de características ocurra en estas capas.

Seguidamente, la salida del último bloque separable es unida mediante una conexión residual, marcada en la figura con una línea intermitente, con la salida concatenada de la segunda capa. Una conexión residual consiste en mezclar las salidas de dos capas de la red neuronal de distinta profundidad, en nuestro caso mediante una operación de suma componente a componente, y aunque es una técnica reciente[14] se ha demostrado que ayuda en gran medida en el entrenamiento de la red en especial en el caso de redes profundas.

A continuación el volumen se separa en dos, cada uno con la mitad de canales, a los que se les aplican otros dos bloques convolucionales. La ventaja de separarlos en dos es una reducción a la mitad del número de parámetros y del tiempo de ejecución. El objetivo de estos bloques es reducir nuevamente el tamaño de los volúmenes para prepararlos para los bloques completamente conectados.

Finalmente, los volúmenes se aplanan resultando en un vector de 32 componentes, que tras una capa completamente conexa con 10 neuronas supone el final de la parte de la red que trata exclusivamente los volúmenes 3D.

La parte de la red que procesa los datos clínicos es menos profunda, siendo simplemente dos bloques completamente conectados que resultan finalmente en un vector de 10 componentes, que se concatena con el vector resultado de la parte anterior, también de 10 componentes, y al que se aplica un último bloque completamente conectado con 4 neuronas. El resultado de este bloque es un vector de 4 componentes, y es lo que denominamos características y por tanto la salida de la subred extractora de las mismas.

En último lugar, dos capas, nótese que no bloques, completamente conectadas toman las características, y tras una función de activación sigmoidal se obtienen valores entre 0 y 1 que son las predicciones para los dos problemas de clasificación.

2.4. DATOS Y PARTICIPANTES

El conjunto de participantes utilizados coincide con el del trabajo original [1], y fue amablemente proporcionado por uno de los autores del artículo. Se compone de 432 hombres y 346 mujeres, de entre 55 y 91 años. El 94% se identificaron como blancos y el 98% como no hispanos. Los participantes están divididos en cuatro clases, una de control con pacientes sanos que no sufren de ninguna enfermedad cognitiva (HC), otra de pacientes de alzhéimer (AD), y por último otras dos de pacientes que en el momento del diagnóstico sufrían de otras enfermedades cognitivas leves (MCI), siendo la primera la de aquellos que en un plazo de tres años desarrollaron alzhéimer (pMCI) y la segunda la de los que no (sMCI).

Los datos utilizados pertenecen a tres tipologías diferentes: datos clínicos, información genética, e imágenes MRI de los cerebros de los pacientes. Todos los datos utilizados son obtenidos al comienzo del seguimiento de los pacientes según el protocolo de ADNI. Estos datos se denominan de *baseline* o *screening*. En la Tabla 2.1 pueden consultarse los detalles sobre dichos datos.

	Nº subjects	Age	F	M	Y in Education	Expresion level			CDRSB	ADAS11	ADAS13	RAVLT			
						0	1	2				immediate	learning	forgetting	% forget
AD	191	75.4±7.5	85	106	15.1±2.9	60	89	42	4.33±1.6	18.6±6.2	28.5±8.6	22.6±7.5	1.7±1.8	4.4±1.9	86.6±25.5
HC	181	74.5±5.8	96	85	16.2±2.7	138	40	3	0.02±0.1	5.5±2.7	8.5±4.1	45.2±10.2	6.2±2.2	3.6±2.7	33.0±27.9
pMCI	179	73.7±7.1	71	108	15.9±2.8	61	88	30	2.0±1.0	13.5±4.3	21.6±5.8	27.3±6.6	2.9±2.2	4.9±2.1	77.8±27.3
sMCI	227	72.3±7.1	94	133	16.0±2.8	144	67	16	1.2±0.7	8.4±3.3	13.5±5.4	38.5±10.1	4.8±2.5	4.4±2.6	49.9±30.6

Tabla 2.1 Resumen de los datos clínicos.

Entre los datos clínicos se incluyen los siguientes datos demográficos: edad, sexo, raza, etnia y los años de educación. Para la raza y etnia, debido a que los pacientes están muy desequilibrados solo distinguimos entre blancos y no blancos e hispanos y no hispanos. También se incluyen los resultados de varios tests cognitivos y de memoria, utilizados comúnmente para el diagnóstico del alzhéimer. En concreto, son los test CDRSB, ADAS11, ADAS13 y RAVLT.

Adicionalmente, se incluye la información genética del APOe4, que se sabe que guarda relación con la enfermedad de Alzheimer. Esta se representa como un valor discreto, {0,1,2}, en función de cuántas copias del gen sufran dicha mutación. Como se trata de un único valor, éste se considera como un elemento más entre los datos clínicos. Nótese que existen tres alelos principales para el mismo gen, APOe2,APOe3 y APOe4, pero en este estudio sólo consideraremos este último.

Todos los datos clínicos fueron estandarizados previamente a utilizarse para el entrenamiento. El proceso de estandarización consiste en, para cada atributo, calcular su media y su desviación estándar, y realizar la siguiente operación:

$$z = \frac{x - \mu}{\sigma}$$

Esto consigue que para todos los atributos estandarizados la media sea $\mu = 0$, y su desviación estándar $\sigma = 1$. El objetivo de la estandarización es, además de ayudar con el proceso de entrenamiento, facilitar la interpretabilidad de los datos, pues los transforma a unidades comparables. Es importante remarcar que la media y desviación estándar han de calcularse sobre el conjunto de datos que se vaya a usar como conjunto de entrenamiento y que los conjuntos de test y validación han de ser estandarizados con esos mismos valores.

Las imágenes MRI son matrices tridimensionales de números reales. Estas requieren de un preprocesado consistente en la reorientación de los volúmenes para coincidir con la orientación del atlas MNI152, un alineamiento afín, un remuestreo respecto al atlas MNI152, y la corrección del *bias field*. Opcionalmente, sobre éstas se puede aplicar una máscara para quitar las partes de la imagen no relevantes para nuestra aplicación, incluyendo el cráneo, cuello y demás estructuras que aparezcan en la imagen no pertenecientes al cerebro. Los detalles de este preprocesado pueden consultarse en el Anexo B.

Adicionalmente, como el rango de los valores de los MRI no es igual en todas las muestras, necesitamos transformarlas para que todas estén en una escala comparable. Para ello estandarizamos las muestras, siguiendo el mismo procedimiento descrito anteriormente, pero individualmente para los valores de cada muestra de manera que tras el proceso todas ellas tengan una media de cero y desviación estándar de uno.

Por último, sobre los determinantes de los jacobianos se realiza un proceso de normalización, escalando su valores entre 0 y 1 utilizando los valores máximo y mínimo del conjunto de entrenamiento.

2.5. ENTRENAMIENTO

Para el proceso de entrenamiento realizaremos un 10-*fold cross validation* para cada experimento realizado, con el fin de obtener métricas del resultado más robustas a error aleatorio. De cada uno de los dos conjuntos de datos, reservaremos en cada *fold* 36 muestras para usarlas como conjunto de validación. Adicionalmente seleccionaremos 32 muestras del problema de clasificación pMCI vs sMCI para utilizar como conjunto de test sobre el que evaluar los modelos entrenados. Los conjuntos de datos de test y validación se eligen aleatoriamente de entre el total de datos, de tal manera que estén equilibrados, es decir que tengan el mismo número de muestras positivas y negativas. No seleccionamos datos de test para el problema AD porque los resultados obtenidos en validación son consistentemente muy parecidos y cercanos al 100% de precisión. Para cada *fold* entrenaremos el modelo durante 40 épocas. Utilizaremos el conjunto de validación para seleccionar la época con menor error de validación, y este será el modelo que se usará para realizar las predicciones de los datos de test.

Los parámetros de entrenamiento que se utilizaron fueron los del artículo donde se propuso la arquitectura[1]. Lo ideal sería realizar una búsqueda exhaustiva para ajustar dichos parámetros, pero esto resultaría prohibitivo debido al gran tiempo que conlleva el entrenamiento del modelo, y no se consideró necesario pues con estos se consiguen resultados de precisión superiores al estado del arte. En concreto, se ha utilizado:

- Un valor para el *dropout* del 0.1 para todas las capas.
- Un coeficiente de la regularización L2 de 5×10^{-5} .
- Un tamaño de *batch* de seis muestras.
- Como función de coste se ha utilizado *binary cross-entropy* para cada uno de los problemas de clasificación. Para el modelo entero se utiliza la suma de ambos costes, multiplicando el del problema de AD vs HC por un factor $\alpha=0.25$ .Esta ponderación en la función de coste viene motivada por considerarse el problema AD vs HC más fácil, como respaldan los resultados previos[1], y porque queremos que la red de prioridad a optimizar la clasificación pMCI vs sMCI.

- Optimizador de Adam , con todos los parámetros iguales a los propuestos por el autor[15], menos la tasa de entrenamiento, que es decreciente según la fórmula :

$$Lr = 0{,}001 * 0{,}3^{\frac{epoca}{10}}$$

- Inicialización de los pesos de las capas convolucionales tal cual se describe en [14].

2.6. Detalles de la implementación

Todo el código del sistema de clasificación desarrollado se ha implementado en Python 3.6. La red neuronal convolucional se construyó mediante el uso de la librería para *deep learning* Keras, versión 2.2.4, y utilizando Tensorflow como *backend*, versión 1.12.

Tensorflow[16] es una biblioteca de cálculo simbólico, gratuita y de código abierto, utilizada habitualmente para diversas aplicaciones de aprendizaje automático, incluyendo redes neuronales. Permite gran flexibilidad a la hora de diseñar modelos y facilita enormemente la ejecución de las mismas en múltiples unidades de procesamiento y en tarjetas gráficas (GPU). Tensorflow viene siendo desarrollado por Google, y usado internamente por esta empresa para producción y investigación.

Keras[17] es una API para el desarrollo de redes neuronales, capaz de utilizar distintas bibliotecas como *backend*, incluyendo Tensorflow, CNTK y Theano. Keras facilita la implementación de modelos de *deep learning* complejos y es fácilmente extensible para la inclusión de módulos customizables, como capas, funciones de coste, optimizadores etc. Además proporciona amplio soporte para la mayoría de componentes comunes en redes convolucionales, incluyendo las variantes de las mismas para convoluciones 3D.

Como las convoluciones separables no estás incluidas en Keras, se ha utilizado la implementación de las mismas proporcionada en github por los autores de la arquitectura original: https://github.com/simeon-spasov/MCI.

El proceso de entrenamiento de todos los experimentos se ha ejecutado sobre una tarjeta gráfica de NVidia Titan RTX, que cuenta con 24 GB de VRAM. Los *drivers* utilizados para ésta incluyen la versión 9.0 de CUDA y la versión 7.5.0 de cuDNN. El resto de especificaciones del equipo utilizado incluyen una RAM de 32 GB y un procesador Intel(R) Core(TM) i7-7700, que dispone de cuatro núcleos a 3,60 Ghz.

Dado que se estaba limitado por el RAM de dicho equipo, se utilizaron números de coma flotante de media precisión (16 bit) para representar los valores de cada voxel de los MRI y determinantes de lo jacobianos utilizados durante los experimentos, de lo contrario se hubiese necesitado acceso a más de 48 GB de RAM.

Para los experimentos que usan como entrada datos clínicos y MRI, el sistema tarda de media unos 25 segundos por época de entrenamiento. Un experimento completo incluyendo los 10 *folds* tarda unas 3 horas. Si se añaden a la entrada del experimento los jacobianos, el sistema tarda de media unos 80 segundos por época. Un experimento completo tarda unas 12 horas. Realizar una predicción de una única muestra tarda, en ambos casos, menos de un segundo.

3. RESULTADOS

3.1. EXPERIMENTOS

Los experimentos realizados en este TFG se han centrado en reproducir los resultados obtenidos por los autores en [1] y analizar la sensibilidad de la red a:

1. Cambios en la naturaleza de los datos de entrada e.g. utilizar sólo datos clínicos, sólo datos de imagen o combinar ambos.
2. Cambios en las imágenes de entrada e.g. trabajar con las imágenes con y sin *skull* utilizando diferentes algoritmos de registro con difeomorfismos.
3. Incluir o no el jacobiano de las transformaciones difeomorfas.

En la Tabla 3.1 se presenta una comparación de todas las métricas calculadas para evaluar los modelos, precisión (*accuracy*, ACC), Área bajo la curva ROC (*area under curve*, AUC), sensibilidad (*sensitivity*, SEN) y especificidad (*specificity*, SPE). Para cada experimento se muestra la mediana de las métricas calculadas en los 10 *folds*. Los valores de las métricas para cada *fold* se calculan en el punto óptimo de la curva ROC, calculado mediante el índice de Youden.

			AD vs HC				pMCI vs sMCI			
Input	Tipo de registro	Skull stripping	ACC	AUC	SEN	SPE	ACC	AUC	SEN	SPE
Clinico	Sin registro		100%	1	100%	100%	84%	0,91	81%	91%
MRI	Sin registro	No	79%	0,8	78%	86%	77%	0,76	81%	75%
	Sin registro	Si	78%	0,8	94%	69%	81%	0,84	75%	78%
	Affine registration	Si	82%	0,84	83%	86%	73%	0,74	88%	66%
	ANTS, SSD	No	76%	0,76	83%	78%	77%	0,78	88%	69%
	ANTS, SSD	Si	78%	0,83	86%	67%	72%	0,71	72%	75%
	BL PDE-LDDMM, SSD	Si	69%	0,73	61%	86%	70%	0,67	66%	72%
	BL PDE-LDDMM, NCC	Si	81%	0,8	78%	89%	75%	0,77	81%	78%
Clinico + MRI	Sin registro	No	100%	1	100%	100%	88%	0,91	88%	94%
	Sin registro	Si	100%	1	100%	100%	88%	0,92	81%	88%
	Affine registration	Si	100%	1	100%	100%	89%	0,94	88%	94%
	ANTS, SSD	No	100%	1	100%	100%	86%	0,92	81%	94%
	ANTS, SSD	Si	100%	1	100%	100%	86%	0,9	84%	88%
	BL PDE-LDDMM, SSD	Si	100%	1	100%	100%	88%	0,9	97%	81%
	BL PDE-LDDMM, NCC	Si	100%	1	100%	100%	89%	0,94	94%	91%
Clinico + MRI + Jac	ANTS, SSD	No	100%	1	100%	100%	84%	0,89	88%	81%
	ANTS, SSD	Si	100%	1	100%	100%	84%	0,89	88%	81%
	BL PDE-LDDMM, SSD	Si	100%	1	100%	100%	88%	0,9	84%	88%
	BL PDE-LDDMM, NCC	Si	100%	1	100%	100%	88%	0,91	88%	88%

Tabla 3.1 Tabla comparativa de los resultados obtenidos.

De la tabla anterior cabe destacar los siguientes resultados:

- Se han conseguido reproducir con casi total exactitud los resultados del trabajo original [1]. En concreto, el experimento con datos Clínico + MRI y el registro con ANTs-SSD ha obtenido los mismos valores de ACC y AUC que en el trabajo original [1]. Esto es importante pues la reproductibilidad de los resultados es de especial interés en este tipo de estudios.

- Los resultados corroboran que el problema de AD vs HC es más fácil de resolver que el de pMCI vs sMCI. Las métricas obtenidas para el primero son casi del 100% en todos los experimentos que incluyen datos clínicos.
- Los mejores resultados conseguidos son usando datos clínicos y MRI como entrada, y BL PDE-LDDMM, NCC para el registro. Además, para todos los casos NCC obtiene mejores resultados que SSD, lo que indica que NCC es mejor métrica para el registro de las imágenes MRI.
- La mejor combinación de datos de entrada es para datos clínicos y MRI. Ninguna de las dos por separado es capaz de conseguir mejores resultados. Adicionalmente, cabe destacar que los resultados no mejoran al añadir los determinantes de los jacobianos como dato de entrada, lo que indica que no parecen aportar más información que la ya proporcionada por los MRI alineados.
- Para los casos con *skull striping* los resultados obtenidos sin registro y con registro afín son similares a los obtenidos con el mejor método de registro (BL PDE-LDDMM, NCC), y mejores que los obtenidos con los registros basados en SSD. Esto indica que un registro menos preciso perjudica al sistema de aprendizaje, pues se pierde más información en el proceso que lo que se gana con la normalización de las imágenes. Aplicar una normalización de las imágenes basada en registro difeomorfo es importante cuando se realizan estudios de morfometría (*voxel based morphometry* o *tensor based morphometry*). En estos estudios se realiza un análisis estadístico de las imágenes que deben estar normalizadas para que el sistema sea capaz de discriminar entre grupos. De acuerdo con los resultados obtenidos en este trabajo, si el sistema de aprendizaje utilizado es lo suficientemente sofisticado es posible que no sea necesaria la normalización, o que sólo sea necesaria una normalización afín. Notar que *tensor based morphometry* sí utiliza la información de los jacobianos para los estudios de morfometría.
- Los resultados obtenidos con el modelo que usa sólo datos clínicos son mejores que los obtenidos con el que usa sólo como entrada los MRI. Esto parece indicar que en el modelo conjunto los datos clínicos tienen más importancia que los MRI, pero es complicado cuantificar en qué medida. No obstante, ambos tipos de datos son necesarios para que el sistema proporcione las máximas prestaciones.

Para cada experimento se generaron diversas gráficas para el análisis en profundidad del funcionamiento del sistema de clasificación. Entre estas gráficas se incluyen un diagrama de caja con las métricas para cada *fold*, una gráfica con los costes de entrenamiento y validación para cada época y gráficas de las curvas ROC para cada uno de los *fold*. A continuación se muestran en las figuras 3.1, 3.2 y 3.3 las gráficas del experimento que obtiene mejores resultados, BL PDE-LDDMM, NCC , así como el que consiguió los mejores resultados en el estudio original[1], ANTS-SSD con *skull striping*. El resto de gráficas pueden observarse en el Anexo E.

Diagramas de caja:

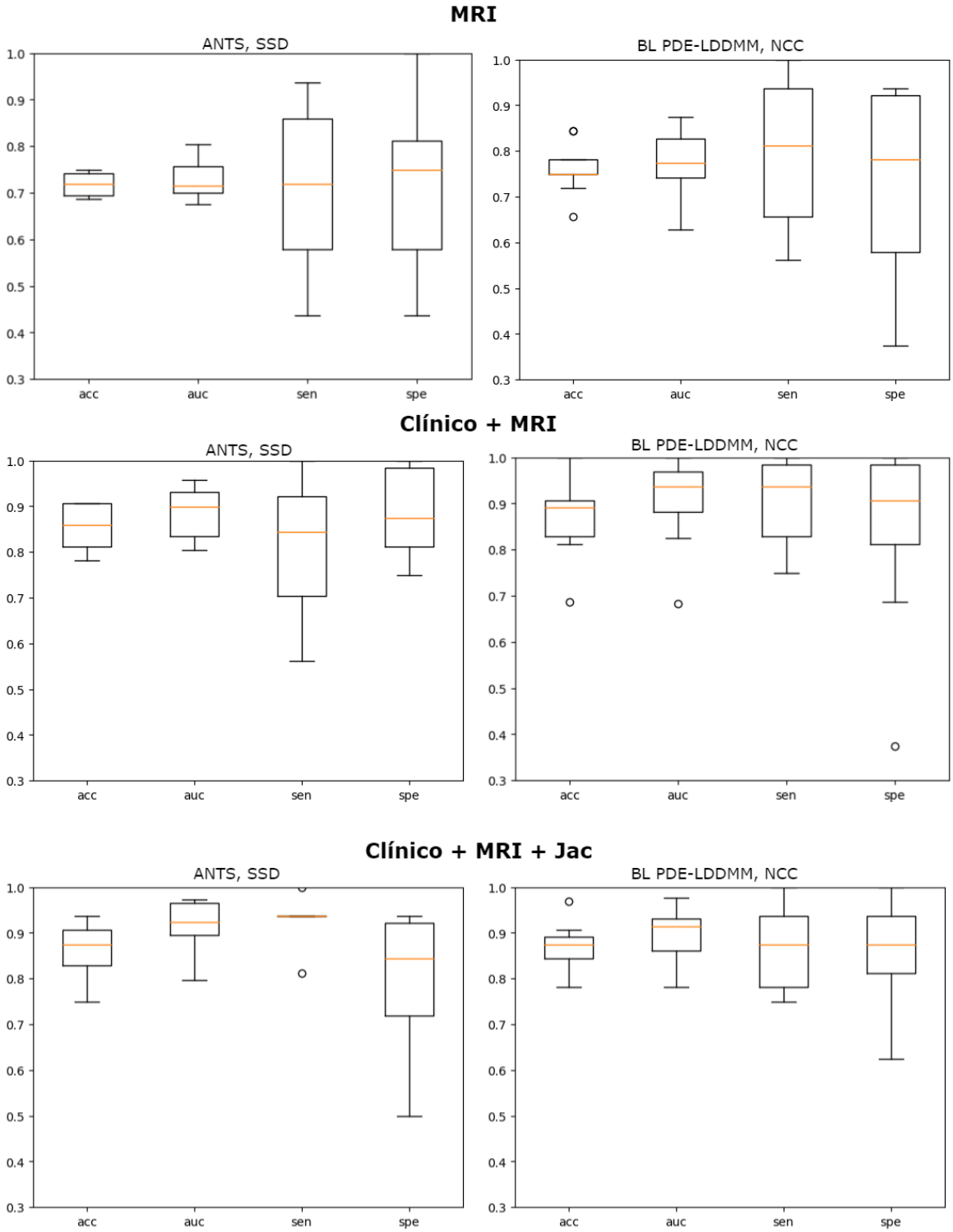


Figura 3.1 Diagramas de caja para los experimentos con skull stripping, datos de entrada MRI (primera fila), Clínico + MRI(segunda fila) y Clínico + MRI + Jac(tercera fila) y registro con ANTS, SSD (columna izquierda) y skull stripping y BL PDE-LDDMM, NCC (columna derecha).

Curvas ROC:

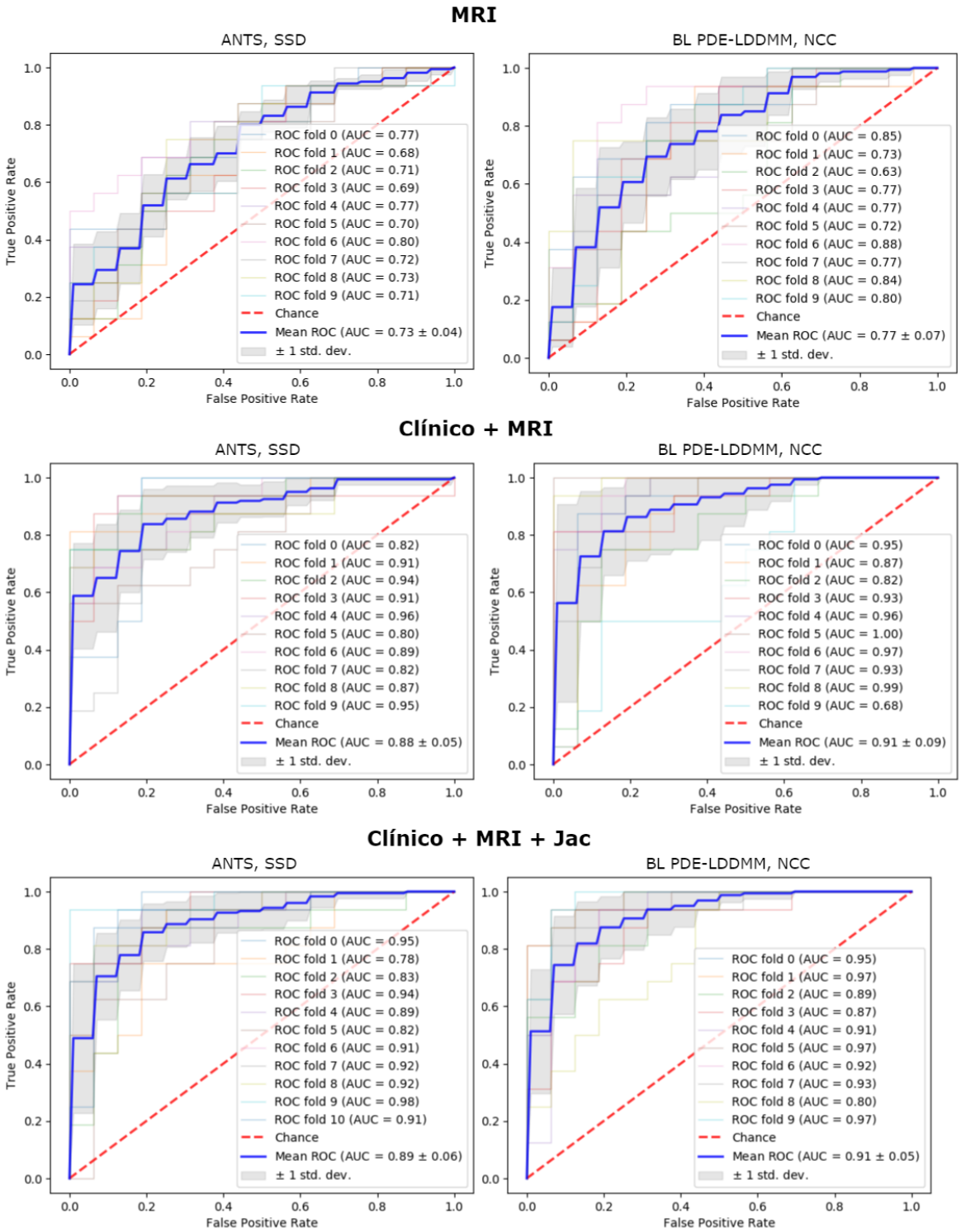


Figura 3.2 ROC para los experimentos con skull stripping, datos de entrada MRI (primera fila), Clínico + MRI(segunda fila) y Clínico + MRI + Jac(tercera fila) y registro con ANTS, SSD (columna izquierda) y skull stripping y BL PDE-LDDMM, NCC (columna derecha). La curva de cada fold viene en un color. También se representa la media de las curvas como una línea gruesa(Azul), y la desviación estándar como la área coloreada alrededor de la misma.

Gráficos de la convergencia de la función de coste:

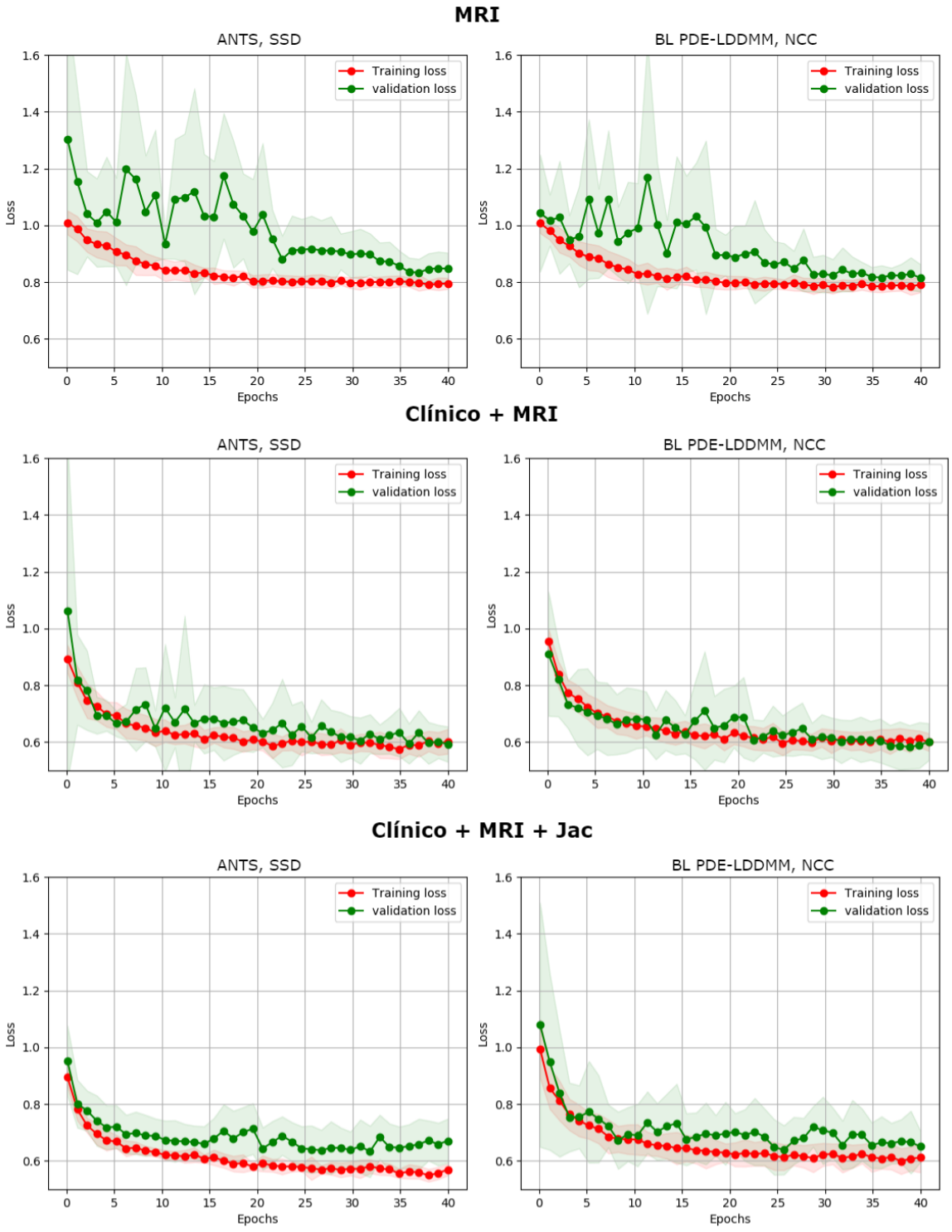


Figura 3.3 Gráfico de la convergencia de la función de coste *para los experimentos con Skull stripping, datos de entrada MRI (primera fila), Clínico + MRI (segunda fila) y Clínico + MRI + Jac (tercera fila) y registro con ANTS, SSD (columna izquierda) y skull stripping y BL PDE-LDDMM, NCC (columna derecha). Se muestra la evolución de los costes de entrenamiento (rojo) y de validación (verde) en función de la época. Las líneas sólidas representan la media entre folds, y las áreas coloreadas la desviación estándar.*

De las gráficas anteriores podemos obtener las siguientes conclusiones:

- Los resultados entre *folds* presentan bastante variabilidad de acuerdo a las métricas utilizadas en los diagramas de caja (figura 3.1), y en las curvas ROC (figura 3.2). Para reducirla, se debería aumentar el tamaño de los grupos de test o bien realizar más *folds* por experimento. Este es un posible punto de mejora en el sistema desarrollado.
- En las gráficas de convergencia de la función de coste (figura 3.3) se puede observar que, a pesar de todas las medidas que se han tomado para evitar el *overfitting,* el modelo lo comete, en especial para los experimentos con solo MRI, y con datos clínicos, MRI y jacobianos. El modelo que regulariza mejor es el que usa solo datos clínicos y MRI, coincidiendo con la combinación de entrada que obtiene mejores resultados.
- En estás gráficas también se puede apreciar el mejor funcionamiento del modelo con PDE-LDDMM, NCC frente a ANTS, SSD. En los diagramas de caja(figura 3.2), puede observarse que además de conseguir una mediana de las métricas mayor, una cantidad significativa de *folds* con NCC obtienen resultados muy buenos, especialmente para Clínico + MRI, mientras que los mejores resultados de SSD están más cerca de la mediana. Esto podría indicar que el incremento esperado en los resultados que conseguiríamos al mejorar el modelo sería mayor para NCC que para SSD. En las curvas ROC (figura 3.3) pueden verse claramente los mejores folds de NCC.

3.2. Análisis de la red neuronal

Un problema común en muchos algoritmos de aprendizaje automático es el de la interpretabilidad de los modelos generados, lo cual consiste en comprender el comportamiento del modelo y en cierta medida explicar las predicciones que realiza. Esto puede facilitar, entre otras cosas, el diseño de modelos con mejores prestaciones, así como la extracción de información relevante al problema, o incluso aumentar la confianza en el correcto funcionamiento del mismo. En esta sección se exploran distintos métodos para interpretar el funcionamiento del modelo utilizado.

3.2.1. Importancia de los datos clínicos

El primer método para valorar la importancia de los datos clínicos en el sistema es el de entrenar la red con un conjunto parcial de los datos de entrada y comparar los resultados obtenidos, normalmente los errores de entrenamiento y validación. Este procedimiento se aplicará sobre los datos clínicos para intentar concluir cuáles son de mayor importancia.

Para ello, se han entrenado varios modelos con los datos clínicos como entrada, durante una única iteración. En primer lugar, se han omitido características una a una, entrenando un modelo para característica omitida. En segundo lugar, se ha entrenado un modelo por cada característica, omitiendo todas las demás . Todos estos modelos se entrenaron y evaluaron con los mismos conjuntos de entrenamiento, validación y test. Los resultados de ambos experimentos pueden verse respectivamente en las tablas 3.2 y 3.3.

	Edad	Genero	Educación	Etnia	Raza	ApoE	CDRSB	ADAS11	ADAS13	R_im	R_le	R_fo	R_%
Error val	0,50	0,51	0,52	0,55	0,59	0,49	0,66	0,66	0,53	0,51	0,54	0,56	0,54
Error train	0,52	0,54	0,53	0,56	0,56	0,52	0,65	0,64	0,54	0,51	0,55	0,59	0,56
ADHC Acc	100%	100%	100%	100%	100%	100%	97%	86%	100%	100%	100%	100%	100%
MCI Acc	88%	84%	88%	81%	84%	88%	81%	78%	81%	84%	88%	81%	81%

Tabla 3.2 Resultados obtenidos tras quitar una característica *de la entrada. Para cada experimento la columna superior indica la característica eliminada del modelo.*

	Edad	Genero	Educación	Etnia	Raza	ApoE	CDRSB	ADAS11	ADAS13	R_im	R_le	R_fo	R_%
Error val	0,85	0,87	0,85	0,93	0,99	0,87	0,62	0,60	0,60	0,63	0,78	0,85	0,72
Error train	0,86	0,85	0,85	0,87	0,87	0,78	0,68	0,64	0,62	0,65	0,71	0,86	0,69
ADHC Acc	69%	53%	72%	50%	50%	75%	100%	100%	97%	94%	86%	67%	97%
MCI Acc	56%	50%	69%	50%	50%	63%	78%	78%	81%	69%	59%	59%	81%

Tabla 3.3 Resultados obtenidos con sólo una característica *como entrada. Para cada experimento la columna superior indica la característica utilizada en el modelo.*

Como puede observarse en la tabla 3.2, Los cambios más significativos sobre el error de validación se producen al eliminar los test cognitivos CDRSB y ADAS11. Adicionalmente, estos son los dos únicos modelos donde no se consigue una precisión del 100% para el problema de AD vs HC. Todo esto indica que son los que tienen un mayor poder predictivo. Esto se corrobora con la tabla 3.3 donde son estos,junto con ADAS13, los que obtienen mejores resultados. Los resultados para el problema de pMCI vs sMCI presentan más dispersión, probablemente debida a error aleatorio al no haberse calculado sobre 10 *folds* para cada experimento sino sobre 1 *fold*.

La tabla 3.3 indica qué atributos tienen un mayor poder predictivo. Se puede observar que todos los test cognitivos obtienen por si solos buenos resultados, especialmente el CDRSB, ADAS11 y ADAS13, con los que se consiguen resultados para la clasificación AD vs HC iguales a los del modelo completo. Esto es debido a que estos test cognitivos son utilizados por los propios médicos en el diagnóstico de alzhéimer, y podría explicar por qué todos los modelos que utilizan los datos clínicos obtienen una precisión de alrededor del 100% para el problema de AD vs HC.

En dicha tabla se puede observar cómo varios atributos apenas consiguen mejorar el 50% de precisión (e.g. género, raza y etnia) que sería el resultado de realizar la clasificación de manera aleatoria. Por último comentar que los factores de riesgo por sí solos, como son la edad, el APoE4 y los años de educación, sí que consiguen superar el 60% para ambos problemas, aunque los resultados obtenidos son bastante peores que los resultados obtenidos por el sistema completo, en especial para el problema de pMCI vs sMCI.

El segundo método para valorar la importancia de los datos clínicos consiste en realizar un análisis de sensibilidad. Este método consiste en calcular la derivada parcial de las predicciones con respecto a cada característica. Cuanto mayor sea el valor obtenido para la derivada parcial respecto a una característica, mayor es la influencia que tiene ésta sobre la predicción final. El cálculo de la derivada parcial de una característica es realizado calculando los valores de la derivada parcial para cada elemento de la muestra y agrupando los resultados como la media de la suma de los cuadrados de los valores obtenidos. El cálculo de la derivada parcial en un elemento de la muestra se realiza fijando el resto de características a un valor concreto y calculando la derivada mediante diferenciación automática, implementada por Tensorflow.

La tabla 3.4. recoge los valores obtenidos para las derivadas de cada característica. De acuerdo a los valores de las derivadas, las dos características que influyen más sobre el resultado de ambos clasificadores son los test cognitivos CDRSB y ADAS13. Aunque en los experimentos de la tabla 3.3. se observó que ADAS11 obtenía por sí solo buenos resultados, parece que no tiene tanta influencia en el

modelo final. Por el contrario, la influencia de APoE4 es considerable siendo, de hecho, superior a la de algunos test cognitivos que obtenían mejores resultados en la tabla 3.3, como ADAS11.

Esto puede ser debido a que la información de los test cognitivos puede resultar redundante para el sistema mientras que APoE aporta información complementaria útil en las tareas de clasificación. La influencia del resto de características, edad ,género, educación, etnia, raza, y test cognitivos RAVLT es similar. Esto no significa que todos ellos tengan el mismo poder descriptivo, sino que el modelo final no es muy sensible a ellos, ya sea porque efectivamente no le resultan útiles al sistema (e.g. raza o etnia) o porque no aportan tanta información como aquellas características a las que el modelo es más sensible.

	Edad	Genero	Educación	Etnia	Raza	ApoE	CDRSB	ADAS11	ADAS13	R_im	R_le	R_fo	R_%
ADHC	0,007	0,007	0,006	0,006	0,005	0,018	0,042	0,013	0,038	0,029	0,011	0,012	0,018
MCI	0,010	0,020	0,010	0,016	0,013	0,040	0,099	0,037	0,121	0,073	0,014	0,009	0,031

Tabla 3.4 Resultados de las derivadas parciales de las predicciones respecto a cada característica. Para cada experimento la columna superior indica la característica utilizada en el modelo.

3.2.2. Análisis de las capas convolucionales

Debido al gran número de parámetros de una capa convolucional, y a la interacción compleja entre las distintas capas, el funcionamiento de las mismas es difícil de analizar. A pesar de esto, debido a la popularización de las CNN en los últimos años, multitud de métodos han sido desarrollados para interpretar el funcionamiento de las mismas. Puesto que las CNN están diseñadas para trabajar con imágenes, muchos de estos métodos buscan representar visualmente algún componente clave de su funcionamiento. Dado que se está trabajando con imágenes en tres dimensiones, esto supone un incremento en la dificultad tanto de visualizar como de interpretar los resultados obtenidos. Todos los métodos de análisis de esta sección se realizaron sobre un modelo entrenado previamente, en concreto uno con solo MRI como datos de entrada, sin registrar y sobre los que se ha aplicado *skull stripping.*

La primera técnica que se va a explorar es la de la visualización de las activaciones de las capas intermedias de la red. Esto puede ayudar a identificar que tipo de información visual conservan las distintas capas, e incluso puede ayudar a identificar si el número de filtros utilizado en alguna capa es demasiado alto, en caso de que repetidamente las activaciones de un filtro sean todo ceros.

En las figuras 3.4 y 3.5 pueden observarse cortes sagitales de las cinco muestras mejor y peor clasificadas del problema pMCI vs sMCI, así como de las activaciones de las primeras tres capas convolucionales. Aunque es difícil extraer conclusiones acerca del motivo por el cual se comete más error al clasificar una muestra que la otra, sí que puede observarse que no es debido a la pérdida de información en alguna capa pues las activaciones de las muestras peor y mejor clasificadas tienen aspecto similar. Cabe destacar que la primera capa mantiene información anatómica de las estructuras cerebrales mientras que, a medida que se profundiza en la red, las activaciones pierden la relación visual con las estructuras anatómicas.

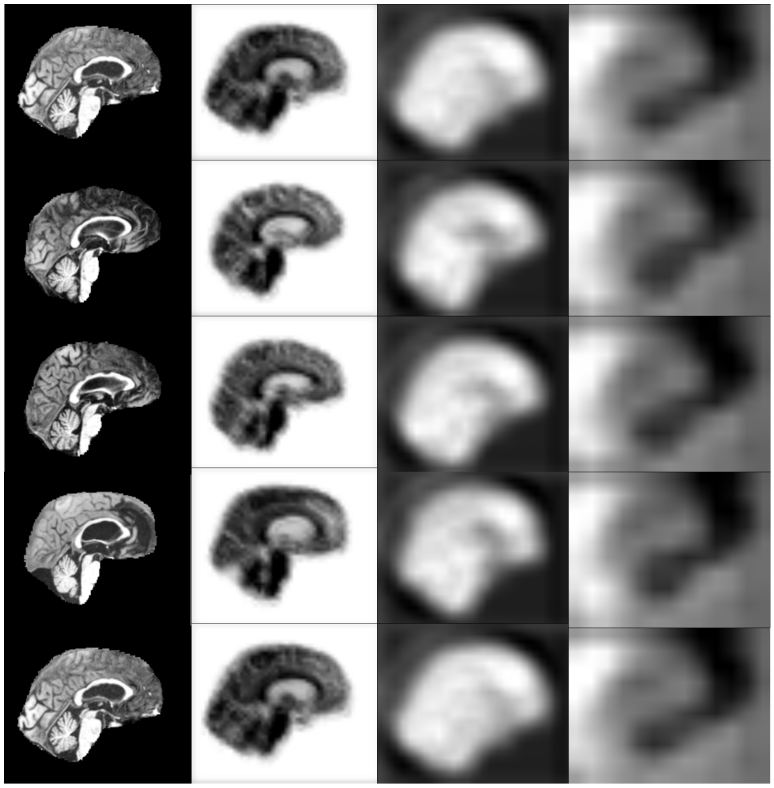


Figura 3.4 Activaciones de las cinco muestras con mayor error.

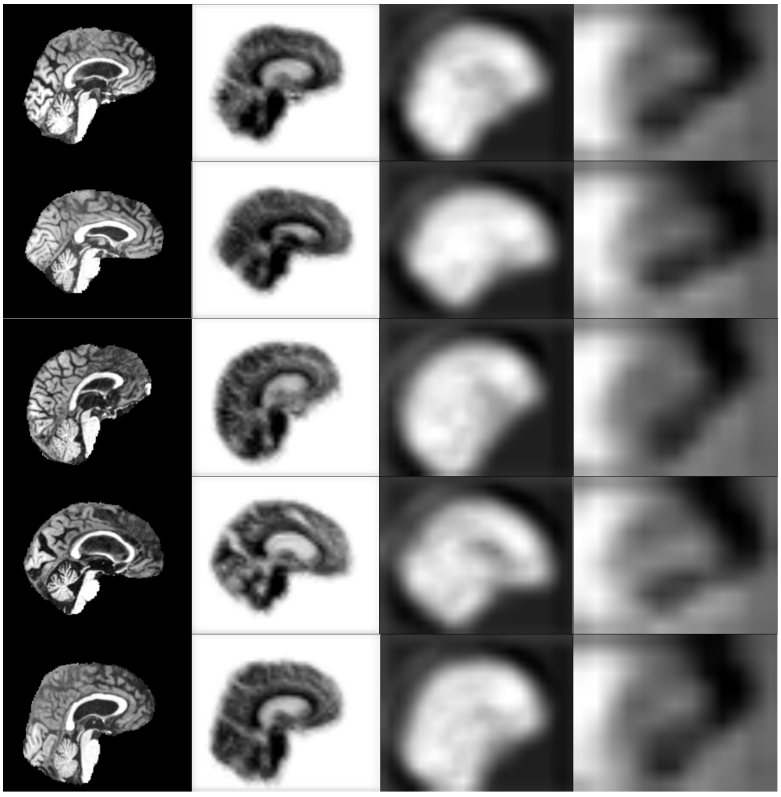


Figura 3.5 Activaciones de las cinco muestras con mayor error.

Una observación importante acerca de las muestras mejor y peor clasificadas es que las cinco mejores son pertenecientes al grupo de MCI estable, mientras que las cinco peores son del grupo que desarrolla alzhéimer en menos de 3 años. Esto indica un posible *bias* del modelo hacia clasificar las muestras como del grupo estable, aunque es difícil concluir cuál es su causa. Es posible que se deba a que las muestras de MCI de los pacientes que desarrollan alzhéimer presentan más diferencias entre sí, y por tanto el clasificador tenga más problemas con ellas.

Otra técnica común de análisis de capas convolucionales es la visualización de los pesos de los filtros de las primeras capas. Como las CNN analizan imágenes, es común que los filtros presenten patrones similares a kernels usados en aplicaciones de visión por computador. Los patrones en las primeras capas suelen parecer conceptos geométricos sencillos, líneas rectas, círculos, etc, y según se profundiza en la red pueden tomar formas más complejas. En la figura 3.6 puede observarse los filtros de la primera capa de una AlexNet entrenada, nótese que la mayoría son patrones de líneas blancas y negras.

Uno de los problemas de la visualización de los filtros es que no en todos los modelos van a obtenerse imágenes interpretables. En concreto, para obtener patrones regulares se necesita una red neuronal entrenada durante muchas épocas, con muchos datos de entrenamiento y que regularice muy bien.

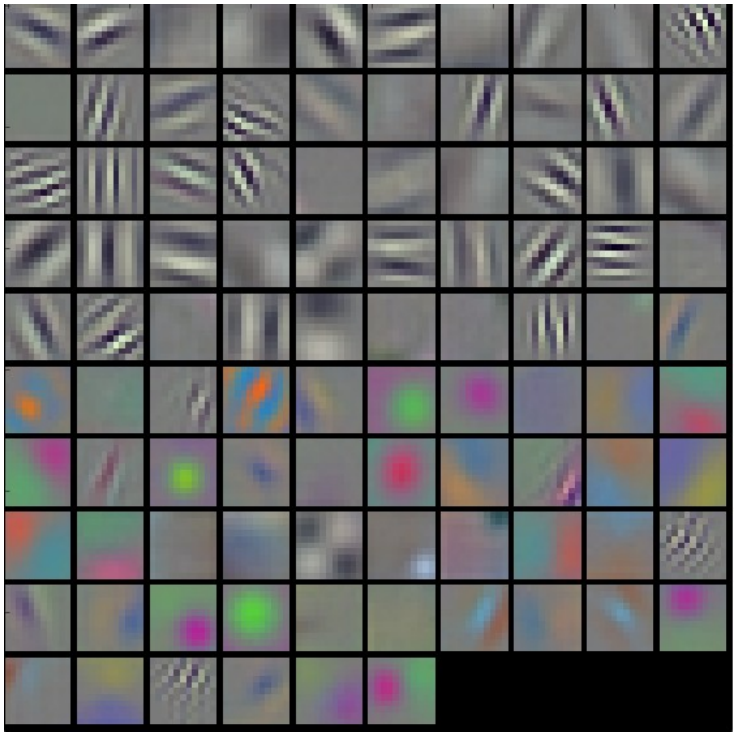


Figura 3.6 Filtros de la primera capa de AlexNet. Imagen obtenida de http://cs231n.github.io/understanding-cnn/.

El la figura 3.6 se pueden ver 13 de los 24 filtros de la primera capa de la red. Los filtros tienen dimensiones 11x11x13 y un único canal, así que cada imagen en una fila es una sección de la última dimensión del filtro, y por tanto cada fila es un filtro completo. Como puede observarse, no hay ningún patrón regular en los filtros de la primera capa, de apariencia similar a los de la figura 3.12. Esto es un indicador más de que a pesar de todos los mecanismos para combatir el *overfitting* que se han utilizado en el modelo se podría mejorar el sistema si se aumentase el número de épocas o el número de muestras.

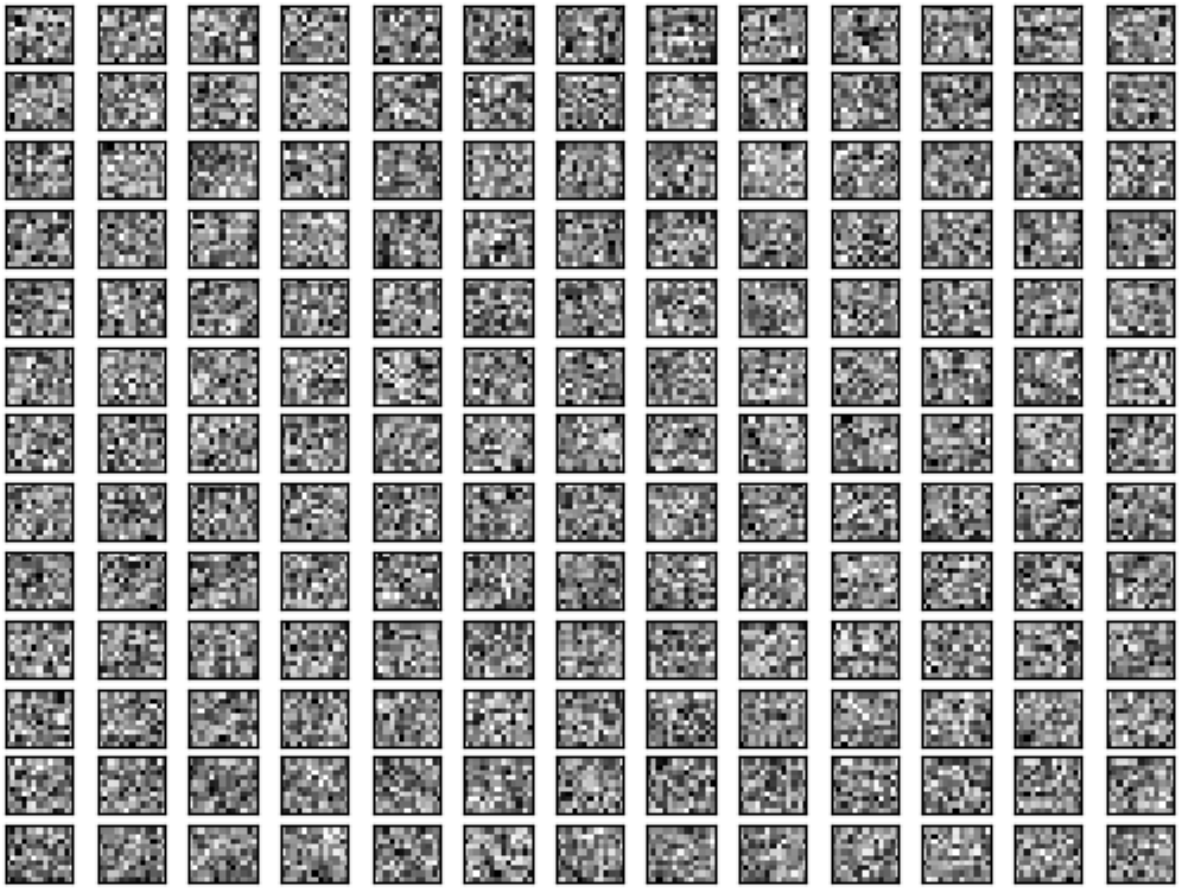


Figura 3.7 Filtros de la primera capa de la CNN utilizada. Cada filtro es de dimensión 11x11x13, de manera que las imágenes de cada fila constituyen un filtro.

4. CONCLUSIONES

El objetivo principal de este TFG ha consistido en desarrollar un sistema basado en redes neuronales convolucionales profundas para abordar dos problemas relacionados con el diagnóstico de la enfermedad de alzhéimer. El primer problema es el de la clasificación de individuos entre pacientes de alzhéimer y miembros del grupo de control. El segundo es el de la predicción de qué pacientes de MCI desarrollarán alzhéimer en un plazo inferior a tres años.

Para llevar a cabo estas tareas, ha sido necesario realizar en primer lugar un estudio del funcionamiento de las redes neuronales convolucionales, así como de los últimos avances realizados en las arquitecturas más modernas. A continuación se ha realizado una implementación del sistema de *deep learning* elegido entre los que han obtenido los mejores resultados en las dos tareas. La implementación se realizó en Python 3.6 mediante el uso de las librerías Keras y Tensorflow. También se ha realizado la selección y descarga de los datos a utilizar en el entrenamiento del repositorio ADNI. Las imágenes MRI se han preprocesado y se han calculado los registros respecto al atlas MNI152 con tres algoritmos de registro difeomorfo diferentes. Finalmente, se realizaron una serie de experimentos que incluyen los descritos en la memoria.

De los resultados obtenidos cabe destacar que se ha conseguido replicar los resultados del trabajo original [1], tanto para el problema de AD vs HC como para el de pMCI vs sMCI. Además, se ha concluido que la combinación de datos de entrada más predictiva es la de datos Clínico + MRI, y que añadir el jacobiano de las transformaciones difeomorfas no mejora los resultados.

Los mejores resultados para el problema de pMCI vs sMCI obtienen un 89% de precisión y un AUC de 0.94. Para el problema de AD vs HC, se obtiene un 100% de precisión y un AUC de 1. Esto es justificable porque el diagnóstico de alzhéimer es una valoración clínica subjetiva, y nuestro sistema utiliza algunos de los datos usados por profesionales médicos para realizar el diagnóstico.

Los resultados obtenidos ponen en duda la necesidad de realizar la normalización de las imágenes utilizando registro no-rígido, como viene siendo necesario para otros sistemas de aprendizaje menos potentes. Este último resultado es de especial relevancia, pues parece estar asumido en la comunidad la necesidad de realizar la normalización de las imágenes para cualquier tipo de estudio similar al realizado en este TFG, siendo el registro difeomorfo una de las partes más costosas de estos sistemas.

Adicionalmente, se realizó un análisis del comportamiento del modelo generado. Para los datos clínicos se concluyó que los datos más descriptivos son los test cognitivos, seguidos de los factores de riesgo genéticos. De las convoluciones se visualizaron los filtros entrenados determinando que el sistema se podría mejorar si se aumentase el número de épocas o el número de muestras, quedando esta mejora como parte del trabajo futuro de este TFG.

En la Figura 4.1. se muestra un diagrama de Gantt con las tareas desarrolladas en la realización del TFG:

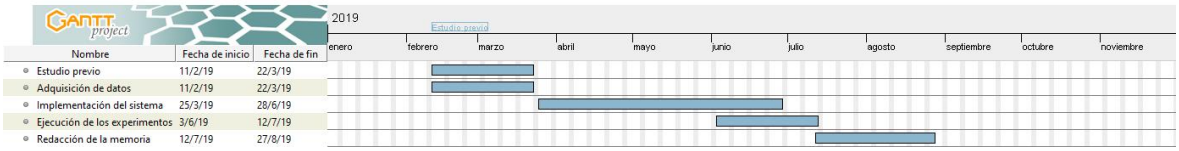


Figura 4.1 Diagrama de gantt del proyecto.

Como líneas futuras de trabajo, se podría aplicar el modelo a otros problemas relacionados con los resueltos en este trabajo. Uno de estos problemas podría ser predecir la evolución de los resultados de los test cognitivos en un plazo de varios años. En iniciativas internacionales como *TadPole challenge* o *Alzheimer's Disease Big Data DREAM Challenge* se han propuesto problemas similares. Sería interesante comprobar qué resultados obtiene nuestro sistema dentro del marco de estos retos.

Con el fin de mejorar el rendimiento del sistema, aumentar el número de muestras sería de especial interés porque, además de ayudar con la regularización del modelo, permitiría explorar el diseño de redes más profundas, que podrían aumentar la capacidad predictiva del sistema. El rendimiento del sistema podría incrementarse mediante la inclusión de otras tareas a realizar dentro de un *multi-task learning* más extenso que podrían ser otros problemas de predicción similares. También podría explorarse la incorporación de técnicas de *transfer learning* que incluyesen información de redes neuronales utilizadas para la segmentación o el registro no-rígido de las imágenes MRI.

Por último creemos que sería interesante incrementar los datos de entrada, incluyendo por ejemplo información genética de los SNPs de los que se sospecha que existe relación con la enfermedad, o cualquier otro biomarcador que se considere que pudiera ser relevante. El sistema podría utilizarse para la búsqueda o validación de biomarcadores, introduciendo por ejemplo información de mutaciones en otros genes y comprobando si aumenta el poder predictivo de la red.

ANEXO A PROYECTO ADNI

Todos los datos utilizados en este proyecto han sido obtenidos de la base de datos de la *Alzheimer's Disease Neuroimaging Initiative* (ADNI). El proyecto ADNI es un estudio longitudinal diseñado para el estudio de la enfermedad de Alzheimer. Su objetivo principal es determinar si la combinación de distintos biomarcadores pueden ser utilizados para medir la progresión del deterioro cognitivo leve (MCI) y realizar el diagnóstico temprano de la enfermedad. El acceso a todos los datos de ADNI es gratis, aunque es necesario realizar una solicitud que es concedida en función de la investigación propuesta. Hasta la fecha, ADNI ha resultado clave en el estudio de la enfermedad de Alzheimer, con más de 1000 publicaciones científicas donde se ha hecho uso de sus datos.

La iniciativa ADNI comenzó en el 2004 como una colaboración entre multitud de empresas públicas y privadas, incluyendo agencias del gobierno de los Estados Unidos como la FDA o grandes farmacéuticas. Los pacientes han sido seleccionados entre ciudadanos de Estados Unidos y Canadá de entre 50 y 90 años. Entre estos se incluyen individuos sanos, que se utilizan como grupo de control, pacientes de alzhéimer y pacientes con deterioro cognitivo leve (MCI). El estudio sigue en activo en la actualidad, y ya cuenta con un total de casi 2000 participantes, divididos a lo largo de cuatro fases de captación de los mismos, ADNI1, ADNIGO, ADNI2 y ADNI3. Cada una de estas fases incluye protocolos actualizados y en ocasiones un aumento de los datos recopilados.

De dichos participantes se realiza un seguimiento de hasta cuatro años, realizando evaluaciones cada aproximadamente tres meses, en las que se recopilan multitud de datos que se consideran podrían servir como biomarcadores de la enfermedad. Entre estos datos encontramos gran variedad de datos clínicos, como datos demográficos, exámenes neurológicos, tests cognitivos e información sobre los posibles medicamentos que estén tomando los participantes. También se incluyen datos genéticos de más de 700.000 SNPs, aunque los que se han demostrado más relevantes hasta la fecha son los relativos a los genes APOe y TOMM40. Además, se toman imágenes de los participantes tanto de tipo MRI como PET, aunque con menos frecuencia que las evaluaciones periódicas. Por último también se recopilan variedad de especímenes biológicos incluyendo muestras de sangre, orina, y líquido cefalorraquídeo. ADNI permite el acceso a los resultados de ciertos análisis realizados sobre dichas muestras, así como a las muestras mismas aunque sólo bajo causas justificadas.

Anexo B Preprocesado de las imágenes MRI

Las imágenes MRI originales de ADNI no son adecuadas para utilizarse directamente como entradas de la red neuronal. Existen múltiples problemas, siendo los más importantes que las imágenes están orientadas de acuerdo a diferentes sistemas de referencia, pueden tener distintos tamaños, se encuentran desalineadas y muestran más o menos parte de zonas del cuerpo que no nos son relevantes para el sistema (e.g. cuello, cráneo etc). Además, todos los cerebros tienen diferentes formas y tamaños, pero están compuestos por las mismas zonas y estructuras. Nos interesa que dichas estructuras aparezcan alineadas aproximadamente sobre las mismas zonas de la imagen, para facilitar su comparación. Para esto es necesario realizar un registro afín que puede refinarse con un registro no-rígido difeomorfo.

En esta sección se detallan los pasos a seguir desde la descarga de las imágenes de ADNI hasta el registro afín (preprocesado). En el siguiente anexo se detallan los métodos de registro no-rígido realizados (normalización).

El preprocesado comienza con la reorientación de las imágenes de acuerdo a la orientación del atlas MNI152 que se considera estándar. Las imágenes MRI no tienen porqué estar orientadas siguiendo esta orientación, es decir, en una imagen los planos axial, sagital y coronal se pueden corresponder con distintos planos respecto a los planos de este atlas. La reorientación de las imágenes se consigue simplemente con rotaciones de 90, 180 o 270 grados respecto a los ejes de la imagen. Este proceso se realizó con la librería FSL[18]. A veces, es necesario realizar una reducción del FOV, es decir, ajustar la altura de la imagen para eliminar posibles partes del cuello que aparezcan en la misma. En la figura B.1 puede verse el mismo plano de dos imágenes, una antes de la reorientación y otra después, y en la figura B.2 puede verse el proceso de reducción de FOV.

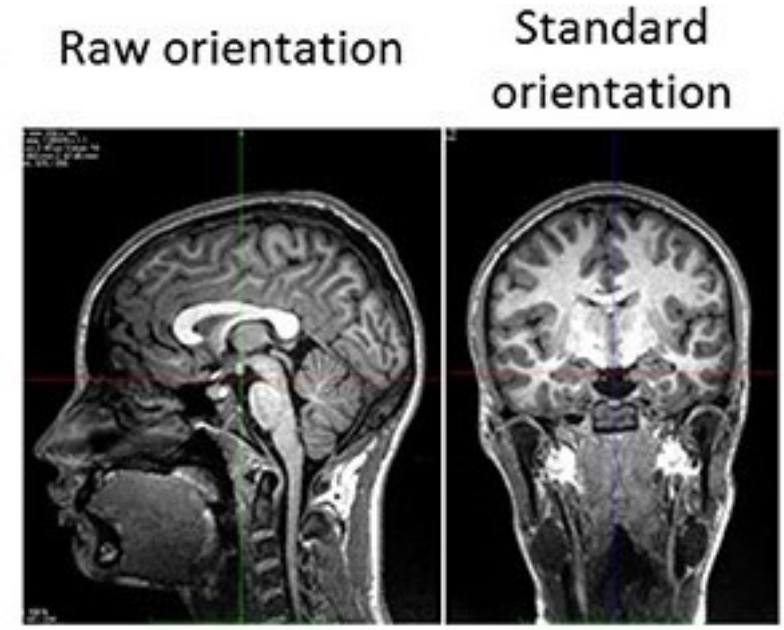


Figura B.1 Reorientación de la vista coronal de un sujeto. Lo que se ve inicialmente como la vista coronal (izquierda) debería ser la vista sagital. Tras la reorientación (derecha) se muestra correctamente la vista coronal. Imagen obtenida de [19].

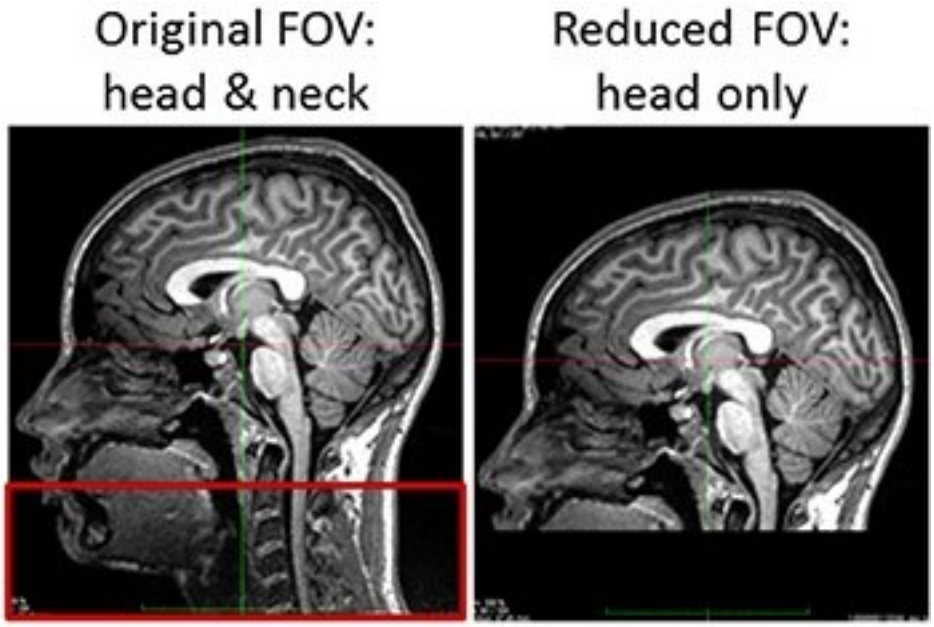


Figura B.2 Reducción del FOV. Sobre la imagen original (derecha) se recorta el volumen rodeado por un rectángulo rojo, para obtener la imagen sin cuello (izquierda). Imagen obtenida de [19].

Después de la reorientación y reducción del FOV las imágenes todavía se encuentran en espacios diferentes, así que el necesario un alineamiento de estas a un espacio común. Esto se consigue mediante la normalización afín de las mismas respecto al atlas MNI152. Para ello se utilizó también la librería FSL. Las imágenes resultantes del proceso de normalización tienen el mismo tamaño que el atlas (182x218x182), y se encuentran alineadas de manera aproximada. En la figura B.3 puede verse una imagen antes y después de la transformación, así como una imagen del atlas MNI152.

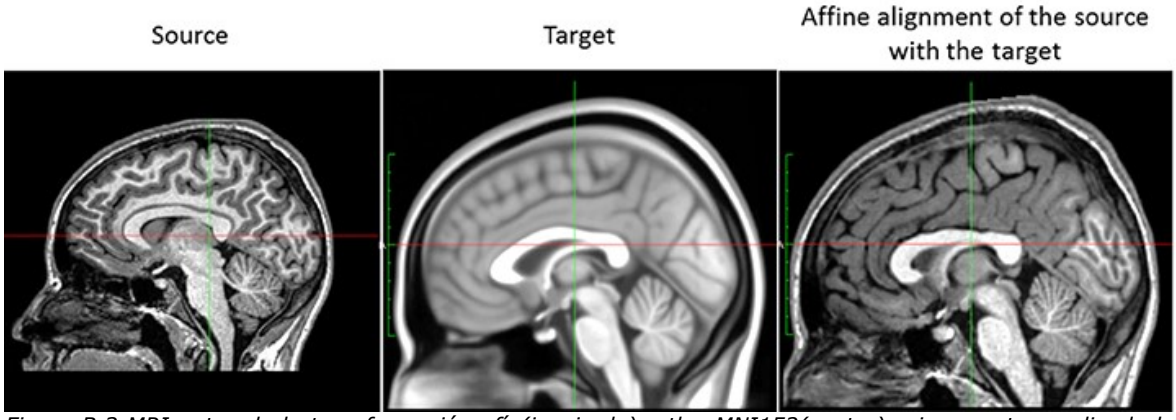


Figura B.3 MRI antes de la transformación afín(izquierda), atlas MNI152(centro) y imagen tras aplicarle la transformación afín(derecha). Imagen obtenida de [19].

Las intensidades de las imágenes MRI poseen una distorsión debido a una señal derivada de aplicar el campo magnético durante la adquisición (*bias field*). Esta señal es de baja frecuencia y suave y se puede eliminar aplicando un algoritmo de corrección. Con esto se consigue que los diferentes tipos de tejidos tengan intensidades similares en la imagen. Así, después de la normalización afín se utilizó el algoritmo de ANTS N3 *bias field correction*[20] para eliminar esta distorsión de las imágenes. En la figura B.4 puede observarse una imagen antes y después del proceso de corrección.

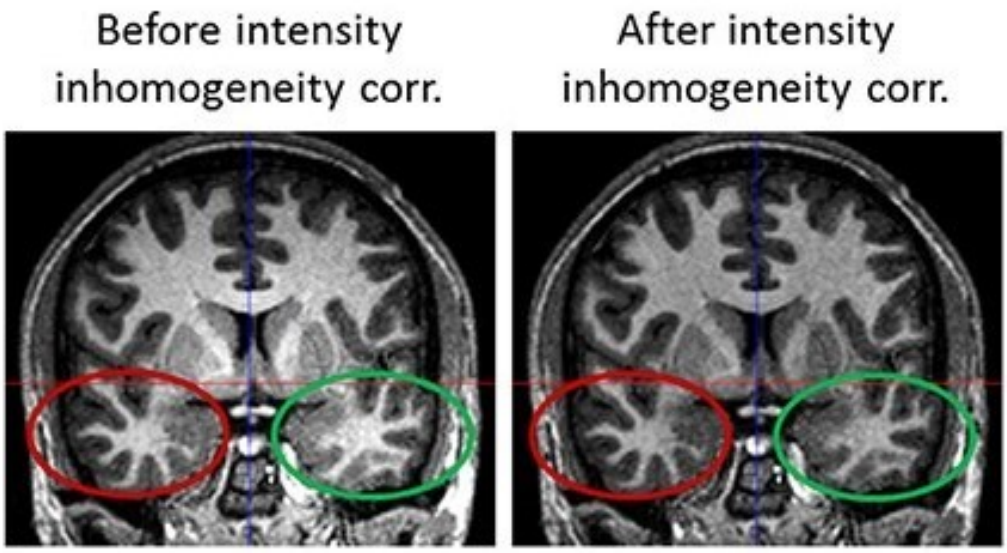


Figura B.4 En la imagen original (izquierda) se puede observar cómo dos zonas del mismo tejido, rodeadas en rojo y verde, presentan intensidades diferentes. Tras la corrección (derecha) ambas zonas presentan intensidades similares. Imagen obtenida de [19].

La última parte del preprocesado consiste en eliminar las estructuras no cerebrales de la imagen (*skull stripping*). Para ello se utilizó la librería Robex[21]. Notar que los experimentos realizados compararon las prestaciones del sistema en imágenes con y sin *skull stripping*. En la figura B.4 puede verse una imagen antes y después de aplicarle *skull stripping*.

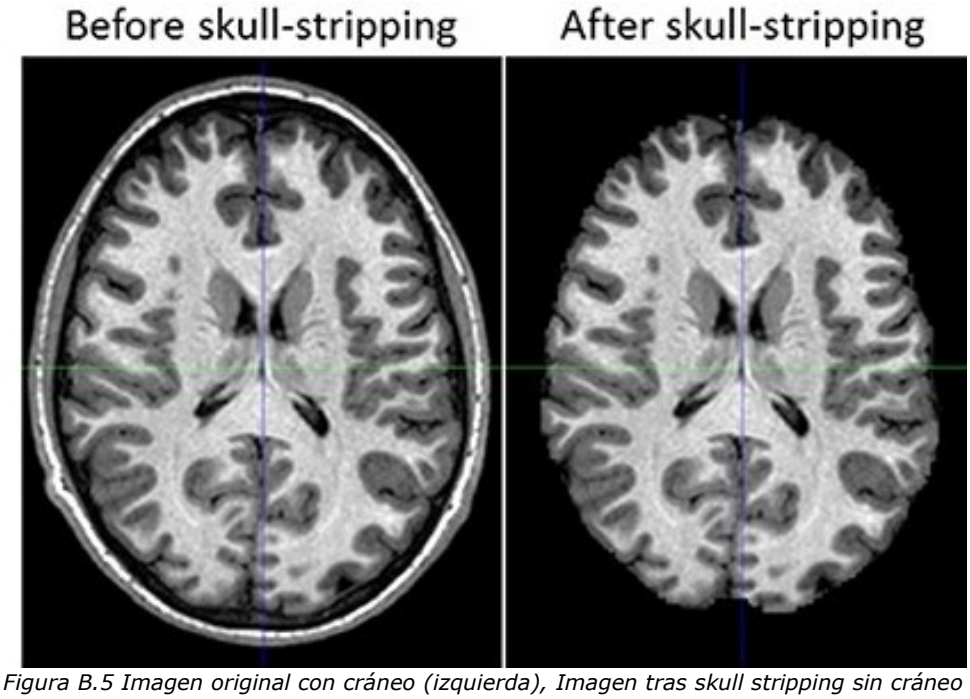


Figura B.5 Imagen original con cráneo (izquierda), Imagen tras skull stripping sin cráneo (centro). Imagen obtenida de [19].

ANEXO C REGISTRO CON DIFEOMORFISMOS

Un difeomorfismo es una aplicación diferenciable, con inversa también diferenciable. En nuestro caso se utilizan en el proceso de normalización para realizar un registro no-rígido suave de las imágenes MRI a el atlas MNI152. La matriz Jacobiana es la matriz de derivadas parciales de la transformación respecto a las coordenadas X, Y, Z y permite medir el grado de expansión o contracción local de la transformación, siendo la matriz identidad la matriz Jacobiana de la transformación identidad. Los difeomorfismos tienen la propiedad de que el determinante de la matriz Jacobiana es positivo, lo cual indica que las transformaciones no se doblan sobre sí mismas (*folding*). Como resultado se conserva la topología de las imágenes transformadas respecto de la original.

En la figura C.1 puede observarse una imagen antes del registro no-rígido, el atlas MNI152 sobre el que se ha registrado y el resultado de dicho registro. Notar la similitud de los ventrículos y el cuerpo calloso y el alineamiento del *cingulate sulcus* después de realizar la normalización.

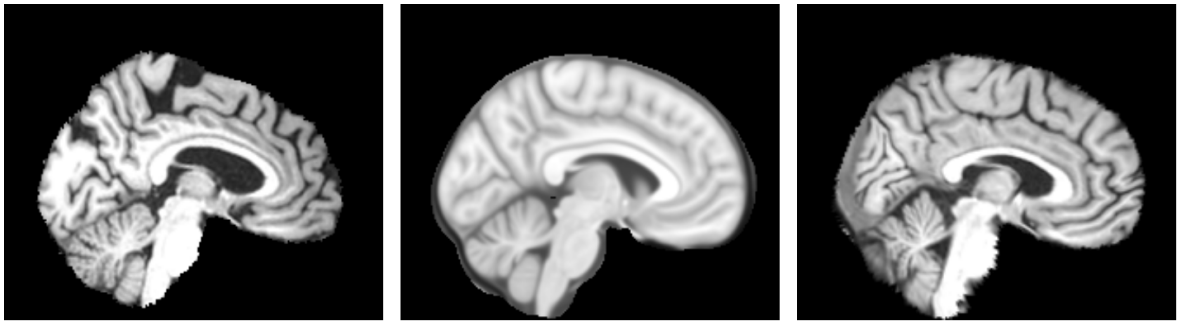


Figura C.1 Imagen original (izquierda), atlas MNI152(centro), imagen normalizada (derecha).

En la figura C.2 pueden verse nuevamente la imagen de la figura C.1. antes y después de la normalización, junto con el determinante de la matriz Jacobiana.

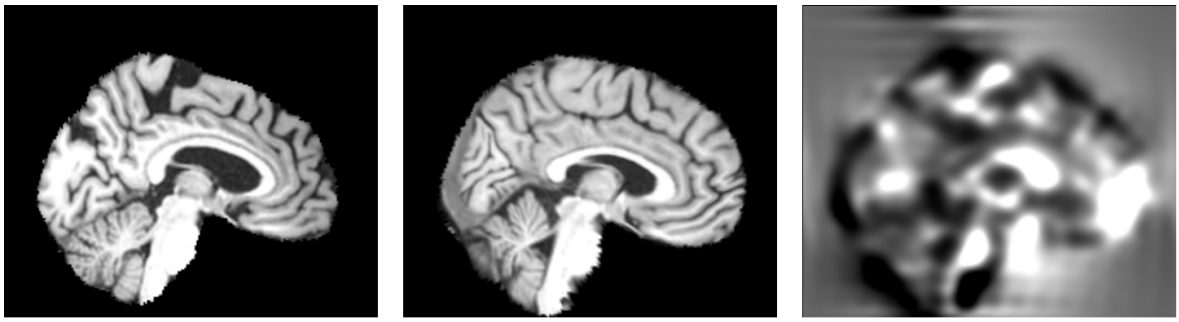


Figura C.2 Imagen original (izquierda), imagen normalizada(centro), determinantes de los jacobianos (derecha).

Existen multitud de métodos para el cálculo del registro con difeomorfismos. En los experimentos realizados en este TFG se han utilizado ANTS[22] y BL PDE-LDDMM [23].

ANEXO D VOXEL BASED MORPHOMETRY Y TENSOR BASED MORPHOMETRY

Voxel based Morphometry (VBM) y *Tensor based morphometry* (TBM) son dos técnicas computacionales usadas en Anatomía Computacional para el estudio de diferencias en concentraciones locales de tejido cerebral. VBM ha sido extensivamente utilizada durante la primera década del siglo XXI para el estudio de poblaciones de pacientes de diferentes patologías cerebrales y estudiar la evolución anatómica de las diferentes regiones cerebrales durante el desarrollo de enfermedades neurológicas y neurodegenerativas.

Ambas técnicas necesitan de la normalización de las imágenes de los pacientes a estudiar en un espacio de referencia común. Esta normalización se comenzó realizando un registro afín que fue reemplazado por un registro no-rígido cuando el desarrollo de estos métodos alcanzó su madurez en el estado del arte. VBM utiliza la imagen normalizada mientras que TBM añade a las imágenes la información de la matriz Jacobiana de la transformación.

VBM realiza un análisis estadístico univariado de las diferencias entre los vóxeles de las imágenes normalizadas, suponiendo que con la normalización se han descartado las diferencias macroscópicas entre la forma de los distintos cerebros, y analizando únicamente los cambios en la estructura local. El análisis estadístico se realiza sobre una función escalar calculada sobre las imágenes normalizadas. Por ejemplo, en [24] se propone utilizar las concentraciones locales de materia gris.

TBM realiza un análisis estadístico multivariado de la matriz Jacobiana de las transformaciones resultantes de la normalización. Se pueden utilizar multitud de medidas derivadas de dicha matriz. La más utilizada es el determinante, pues permite medir el cambio de volumen relativo entre la imagen del paciente y la imagen utilizada como referencia.

ANEXO E RESULTADOS EXPERIMENTALES

Durante el transcurso del trabajo, se han realizado un elevado número de experimentos, de los cuales solo se han discutido los más relevantes en la memoria. En este anexo se presentarán las métricas obtenidas así como las gráficas de todos ellos.

Los experimentos se ordenarán por la combinación de datos de entrada que utilicen, y para cada uno se indicará si se ha realizado skull stripping sobre los MRI y el tipo de registro.

Algunos experimentos se realizaron para evaluar la mejor forma de estandarizar los datos MRI, lo cual se indicará a continuación junto con demás comentarios. Aquellos experimentos donde no se indique serán los que se han estandarizado tal cual se describe en 2.4, lo cual denominaremos estandarización por muestra, y es la utilizada por defecto en todos los experimentos previamente descritos.

Para cada experimento se mostrarán la mediana de las métricas obtenidas en los 10 *folds* , un gráfico de la convergencia de la función de coste, un diagrama de caja para el problema pMCI vs sMCI, y un gráfico con las curvas ROC de cada fold para el problema de AD vs HC y otro para el problema de pMCI vs sMCI.

Datos de entrada: Clínico

Experimento *1* *:* Sin registro

	ACC	AUC	SEN	SPE
AD vs HC	100%	1.00	100%	100%
pMCI vs sMCI	88%	0.90	97%	81%

Tabla E.1: Métricas obtenidas por el experimento 1.

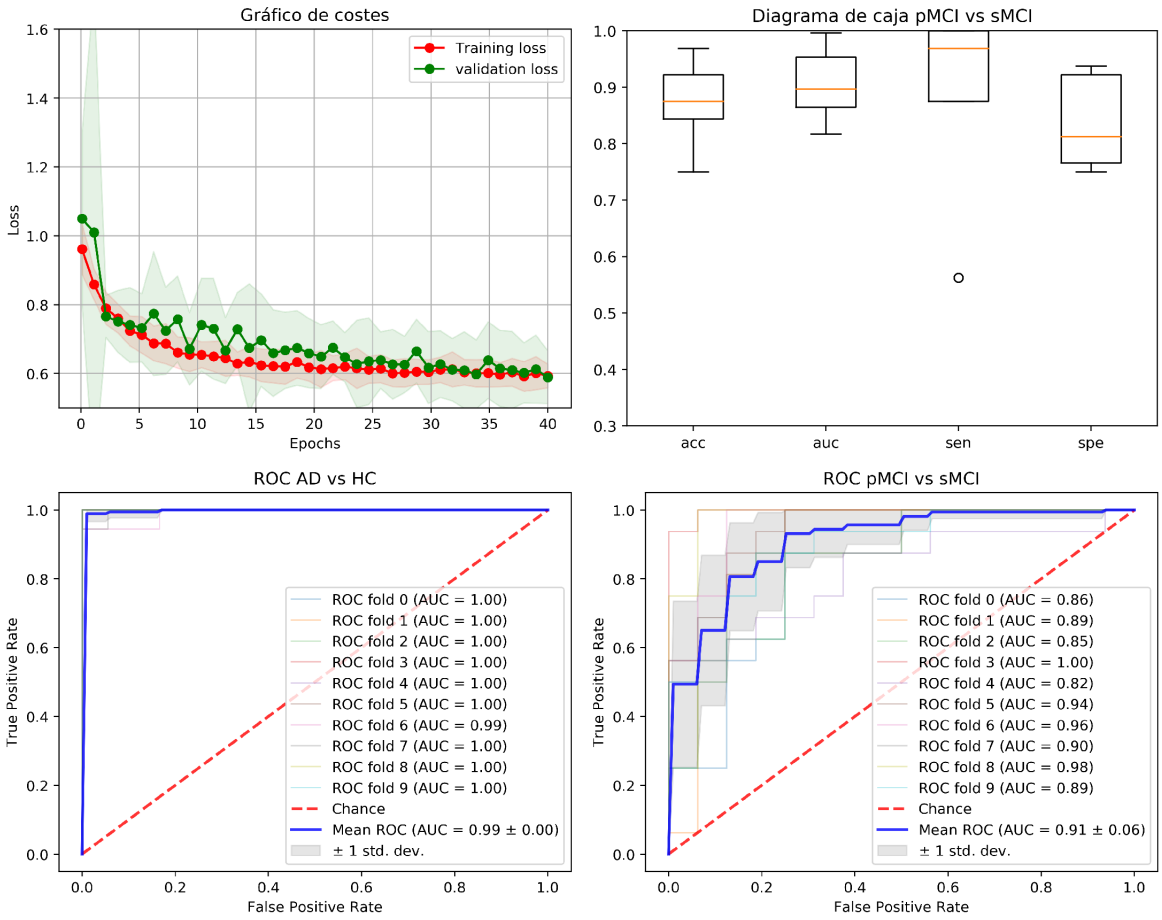


Figura E.1: Gráficas obtenidas por el experimento 1. En la esquina superior izquierda un gráfico de convergencia de la función de coste, en la esquina superior derecha un diagrama de cajas para el problema pMCI vs sMCI, en la esquina inferior izquierda una curva ROC para el problema de AD vs HC y en la esquina inferior derecha una curva ROC para el problema de pMCI vs sMCI.

Datos de entrada: MRI

Experimento *2 :* Sin registro

	ACC	AUC	SEN	SPE
AD vs HC	79%	0.80	78%	86%
pMCI vs sMCI	77%	0.76	81%	75%

Tabla E.2: Métricas obtenidas por el experimento 2.

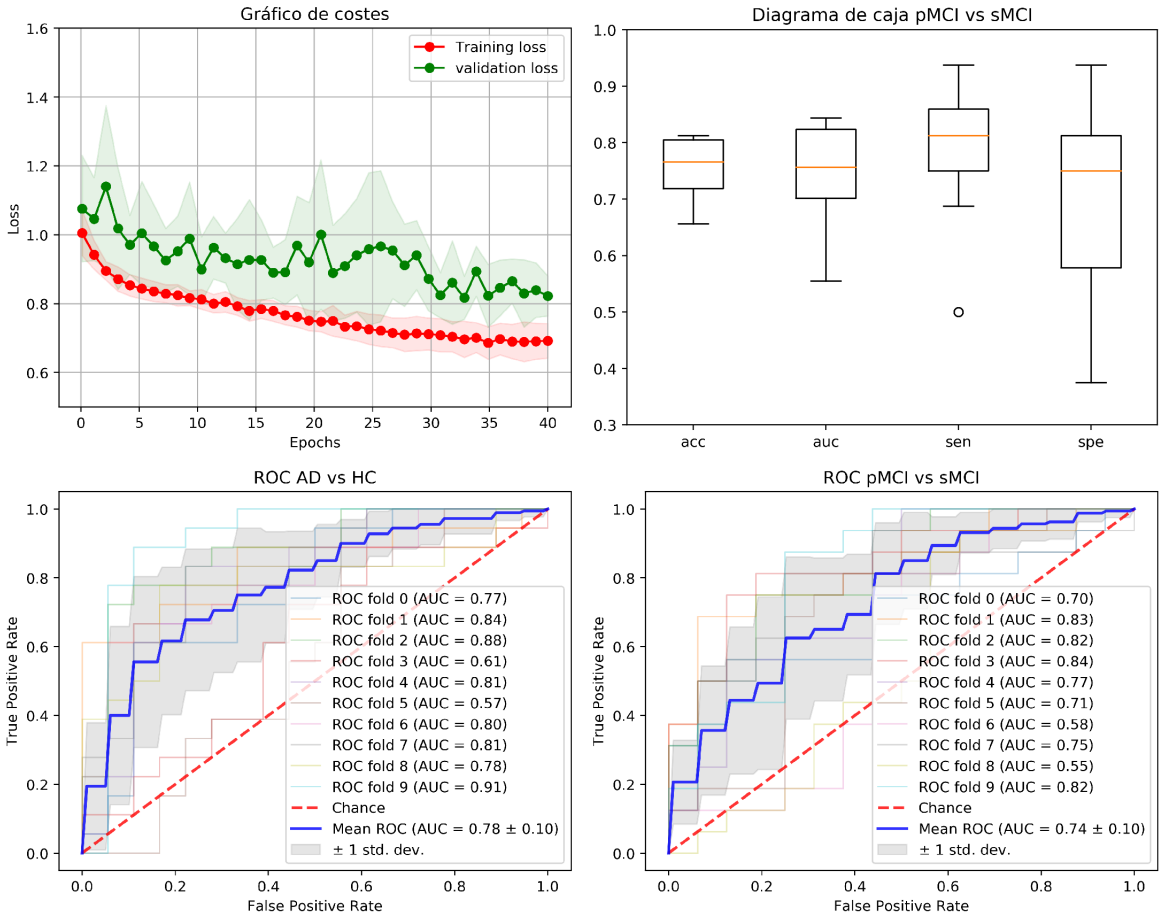


Figura E.2: Gráficas obtenidas por el experimento 2. En la esquina superior izquierda un gráfico de convergencia de la función de coste, en la esquina superior derecha un diagrama de cajas para el problema pMCI vs sMCI, en la esquina inferior izquierda una curva ROC para el problema de AD vs HC y en la esquina inferior derecha una curva ROC para el problema de pMCI vs sMCI.

Experimento *3* : Sin registro,*skull stripping*

	ACC	AUC	SEN	SPE
AD vs HC	78%	0.80	94%	69%
pMCI vs sMCI	81%	0.84	75%	78%

Tabla E.3: Métricas obtenidas por el experimento 3.

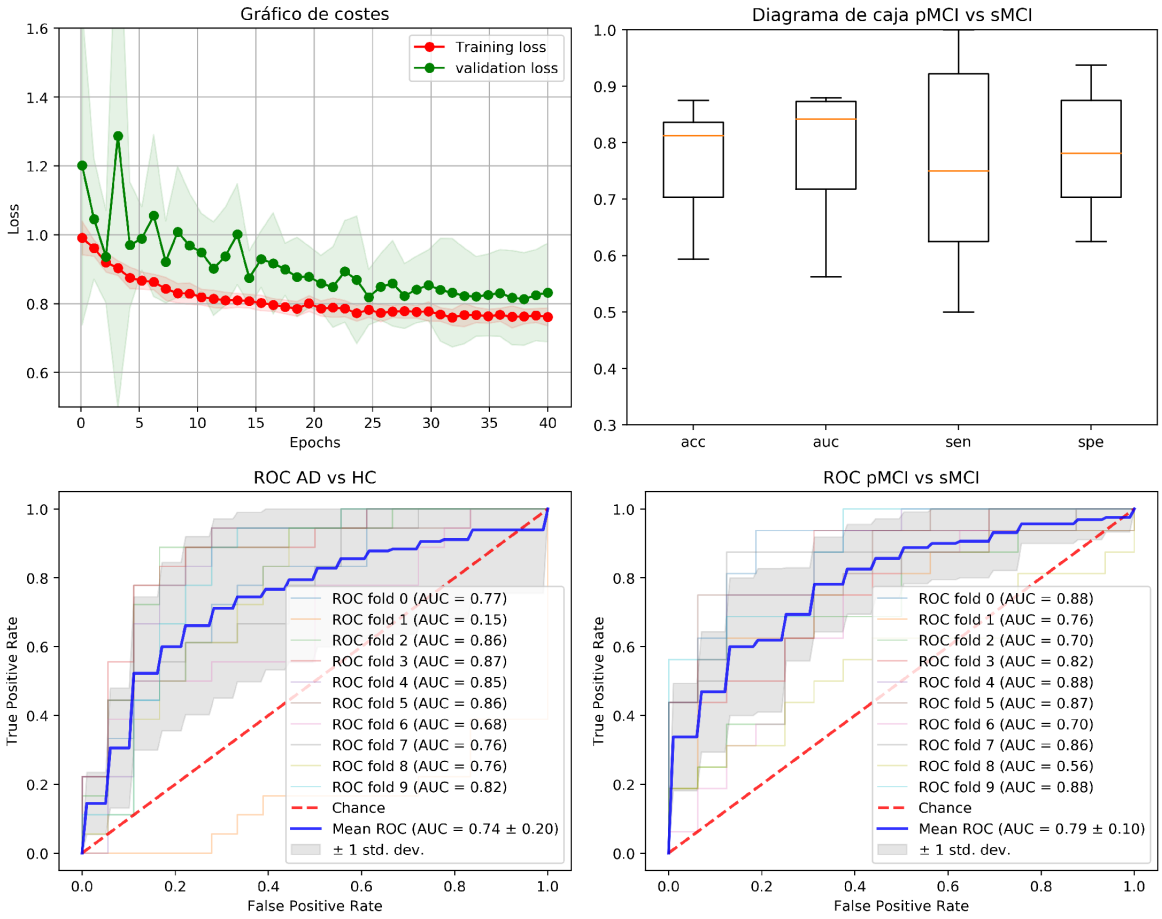


Figura E.3: Gráficas obtenidas por el experimento 3. En la esquina superior izquierda un gráfico de convergencia de la función de coste, en la esquina superior derecha un diagrama de cajas para el problema pMCI vs sMCI, en la esquina inferior izquierda una curva ROC para el problema de AD vs HC y en la esquina inferior derecha una curva ROC para el problema de pMCI vs sMCI.

Experimento *4* : *Affine registration,skull stripping*

	ACC	AUC	SEN	SPE
AD vs HC	82%	0.84	83%	86%
pMCI vs sMCI	73%	0.74	88%	66%

Tabla E.4: Métricas obtenidas por el experimento 4.

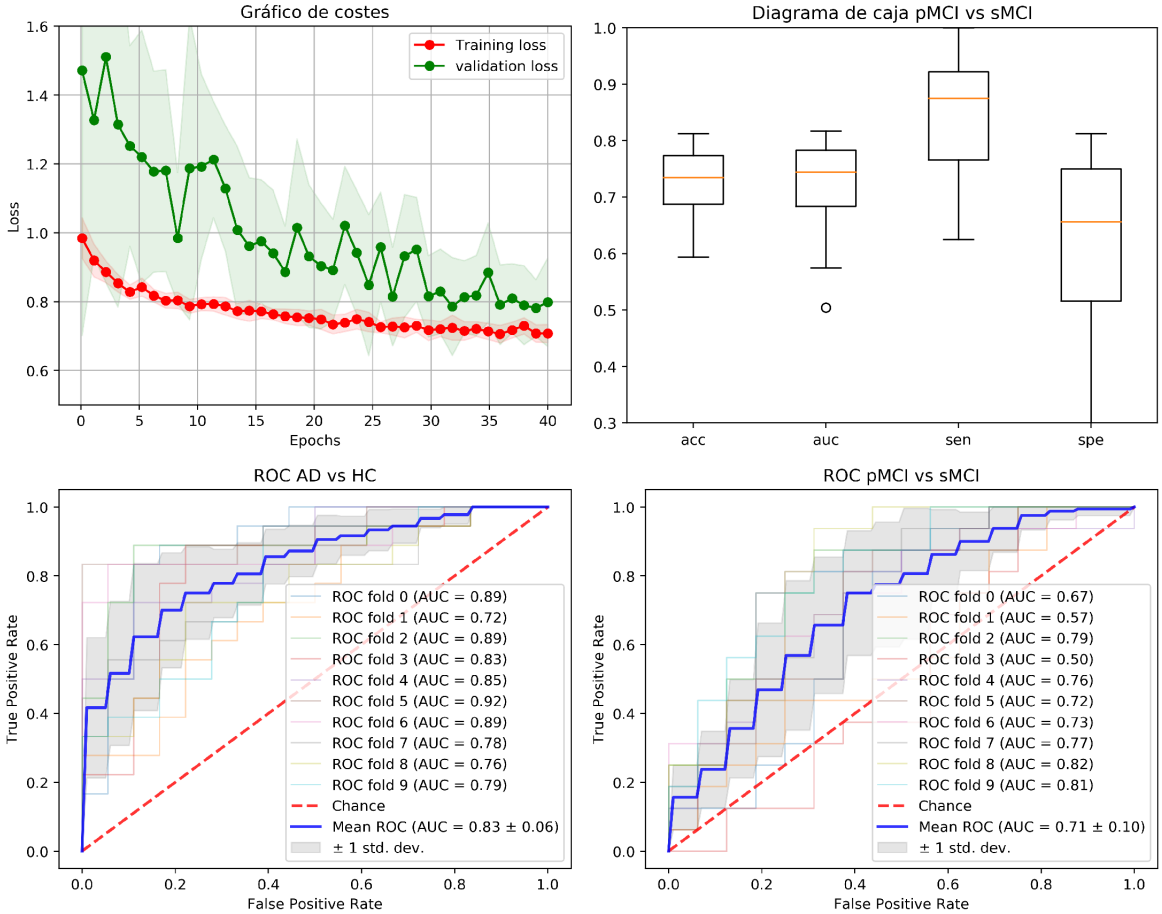


Figura E.4: Gráficas obtenidas por el experimento 4. En la esquina superior izquierda un gráfico de convergencia de la función de coste, en la esquina superior derecha un diagrama de cajas para el problema pMCI vs sMCI, en la esquina inferior izquierda una curva ROC para el problema de AD vs HC y en la esquina inferior derecha una curva ROC para el problema de pMCI vs sMCI.

Experimento *5* : *Affine registration,skull stripping*

Con estandarización por muestra y luego normalización por característica.

	ACC	AUC	SEN	SPE
AD vs HC	78%	0.81	83%	75%
pMCI vs sMCI	75%	0.78	78%	78%

Tabla E.5: Métricas obtenidas por el experimento 5.

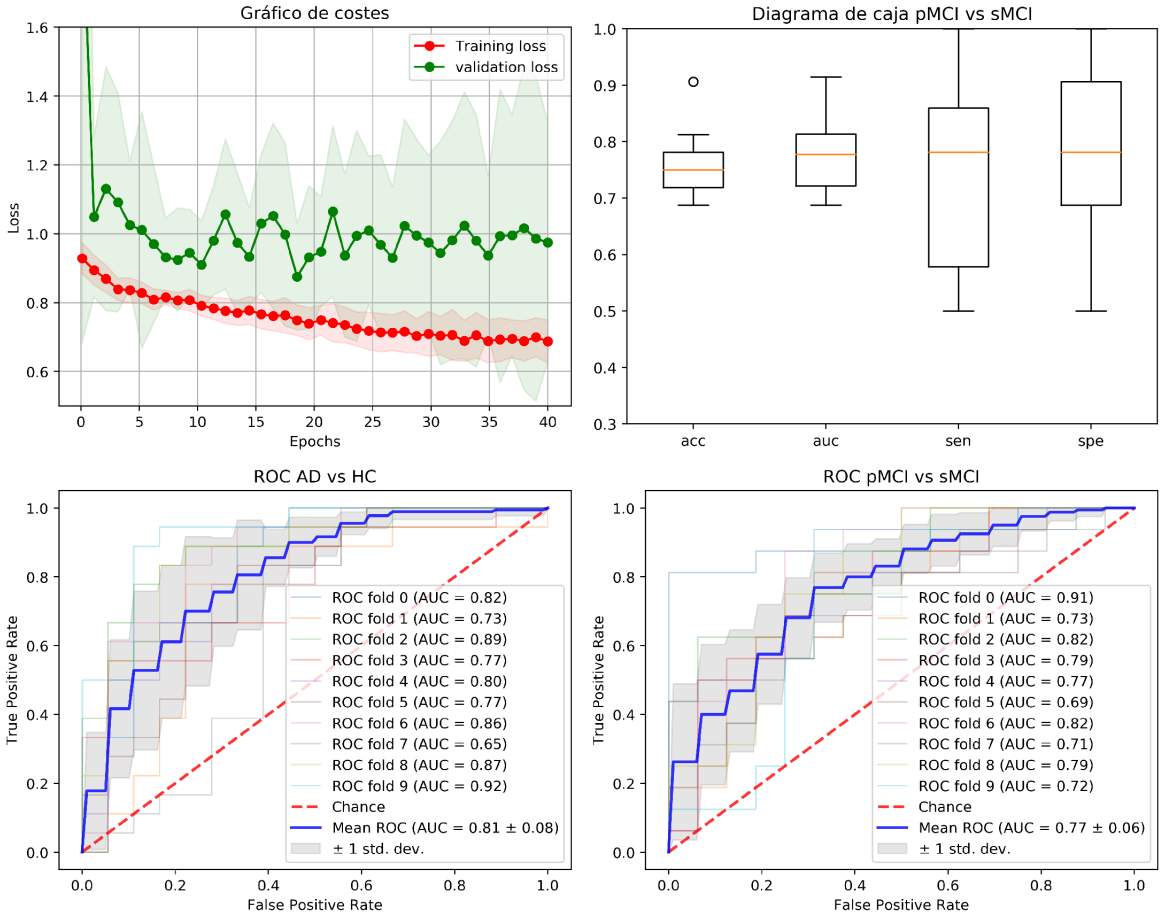


Figura E.5: Gráficas obtenidas por el experimento 5. En la esquina superior izquierda un gráfico de convergencia de la función de coste, en la esquina superior derecha un diagrama de cajas para el problema pMCI vs sMCI, en la esquina inferior izquierda una curva ROC para el problema de AD vs HC y en la esquina inferior derecha una curva ROC para el problema de pMCI vs sMCI.

Experimento 6 : *Affine registration,skull stripping*

Con estandarización por muestra y luego normalización por característica.

	ACC	AUC	SEN	SPE
AD vs HC	76%	0.79	69%	86%
pMCI vs sMCI	67%	0.66	62%	78%

Tabla E.6: Métricas obtenidas por el experimento 6.

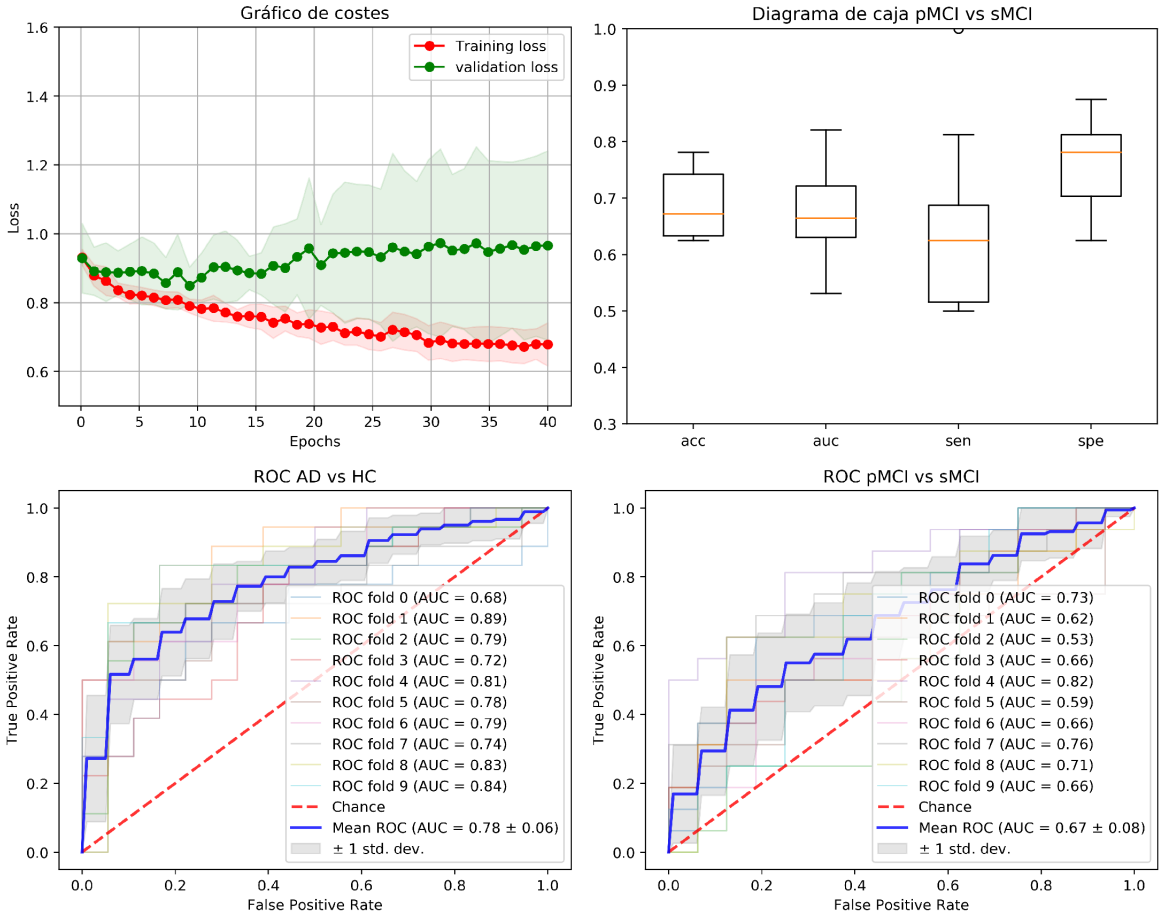


Figura E.6: Gráficas obtenidas por el experimento 6. En la esquina superior izquierda un gráfico de convergencia de la función de coste, en la esquina superior derecha un diagrama de cajas para el problema pMCI vs sMCI, en la esquina inferior izquierda una curva ROC para el problema de AD vs HC y en la esquina inferior derecha una curva ROC para el problema de pMCI vs sMCI.

Experimento *7* : *Affine registration,skull stripping*

Con estandarización por característica y mascara mínima tras registro.

	ACC	AUC	SEN	SPE
AD vs HC	68%	0.67	61%	64%
pMCI vs sMCI	73%	0.76	81%	69%

Tabla E.7: Métricas obtenidas por el experimento 7.

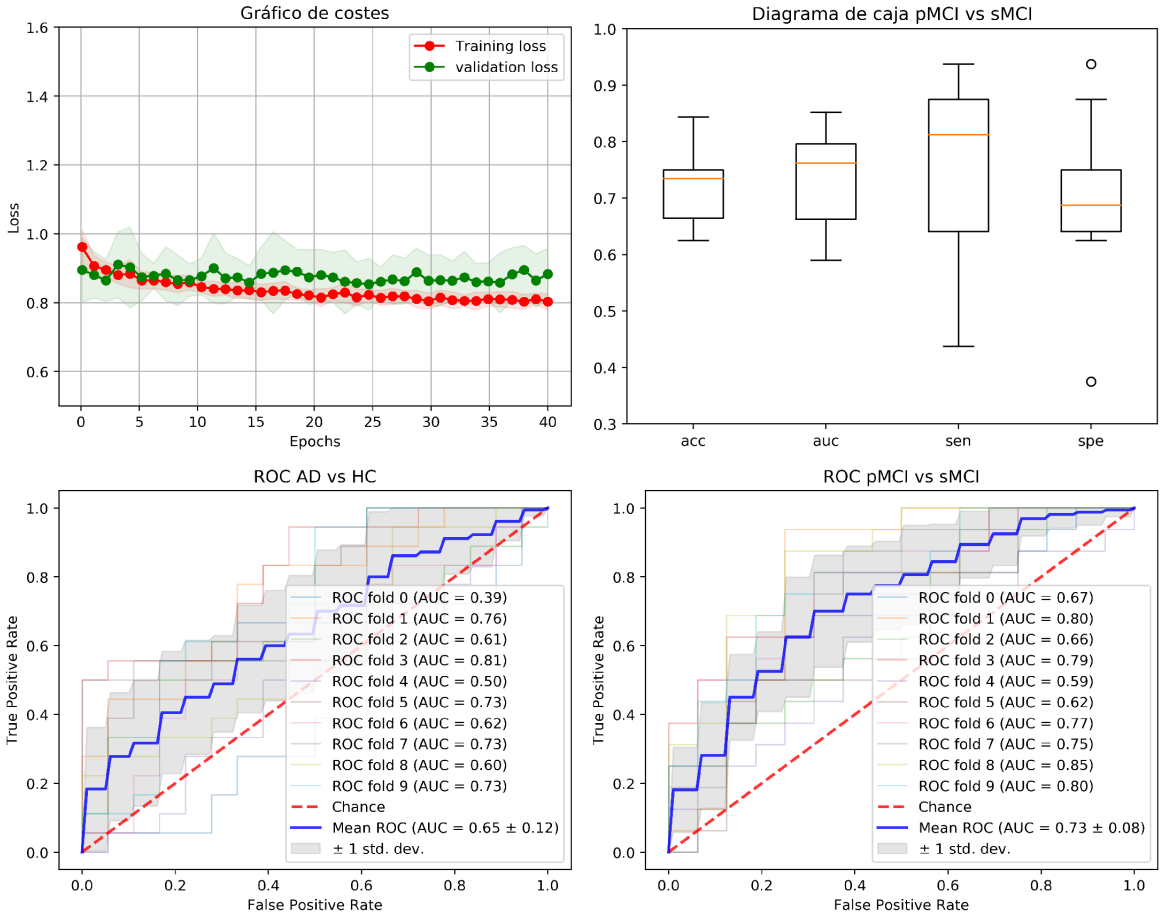


Figura E.7: Gráficas obtenidas por el experimento 7. En la esquina superior izquierda un gráfico de convergencia de la función de coste, en la esquina superior derecha un diagrama de cajas para el problema pMCI vs sMCI, en la esquina inferior izquierda una curva ROC para el problema de AD vs HC y en la esquina inferior derecha una curva ROC para el problema de pMCI vs sMCI.

Experimento *8 :* BL PDE-LDDMM, NCC,*skull stripping*

	ACC	AUC	SEN	SPE
AD vs HC	81%	0.80	78%	89%
pMCI vs sMCI	75%	0.77	81%	78%

Tabla E.8: Métricas obtenidas por el experimento 8.

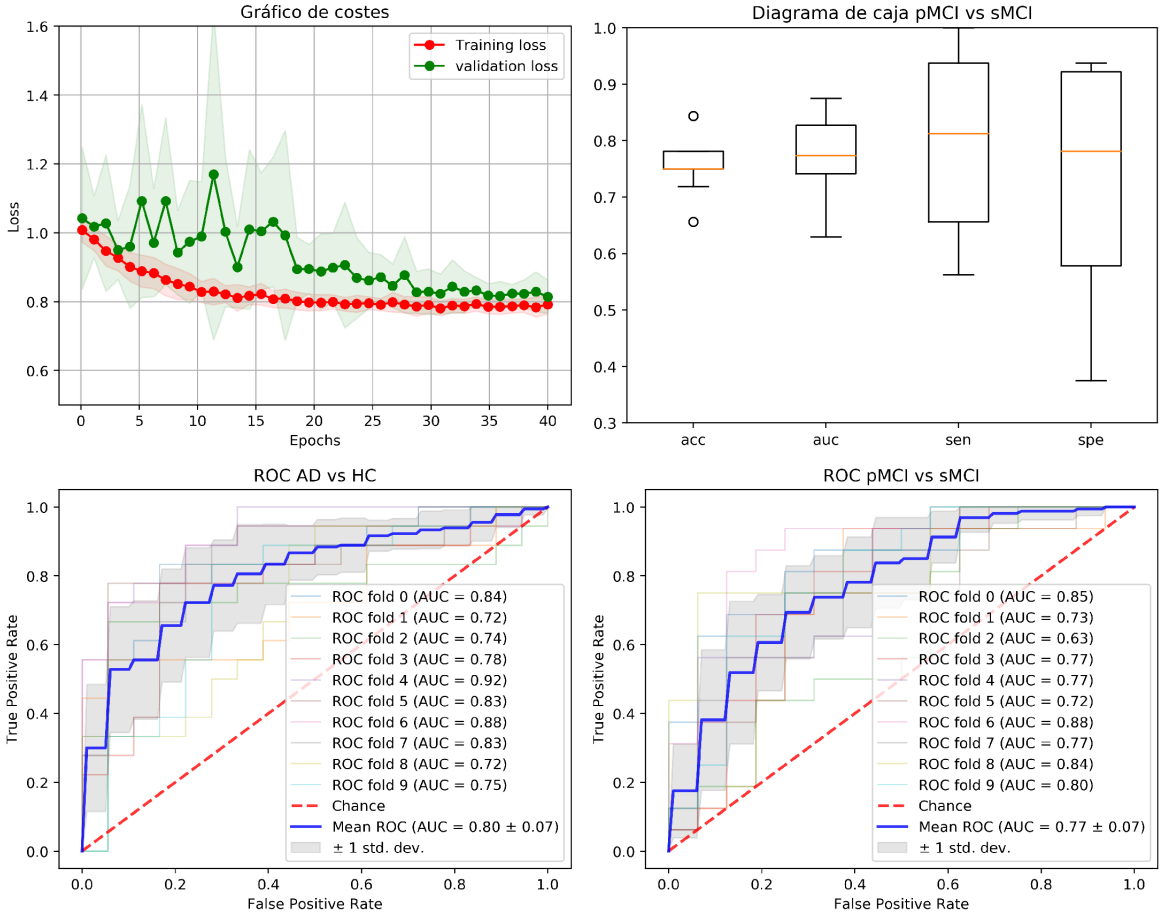


Figura E.8: Gráficas obtenidas por el experimento 8. En la esquina superior izquierda un gráfico de convergencia de la función de coste, en la esquina superior derecha un diagrama de cajas para el problema pMCI vs sMCI, en la esquina inferior izquierda una curva ROC para el problema de AD vs HC y en la esquina inferior derecha una curva ROC para el problema de pMCI vs sMCI.

Experimento 9 : ANTS,SSD

Con estandarización por característica.

	ACC	AUC	SEN	SPE
AD vs HC	64%	0.55	64%	69%
pMCI vs sMCI	67%	0.69	75%	72%

Tabla E.9: Métricas obtenidas por el experimento 9.

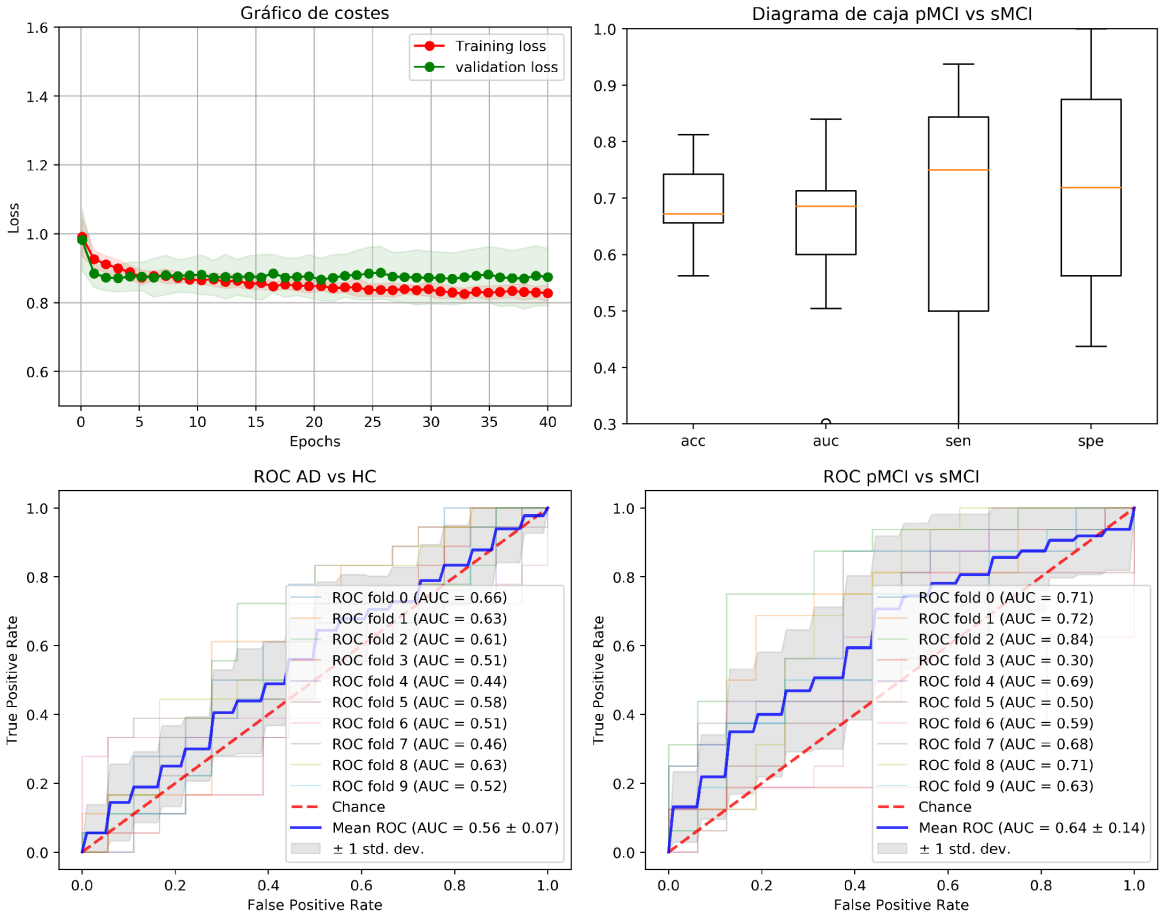


Figura E.9: Gráficas obtenidas por el experimento 9. En la esquina superior izquierda un gráfico de convergencia de la función de coste, en la esquina superior derecha un diagrama de cajas para el problema pMCI vs sMCI, en la esquina inferior izquierda una curva ROC para el problema de AD vs HC y en la esquina inferior derecha una curva ROC para el problema de pMCI vs sMCI.

Experimento *10* : ANTS,SSD

	ACC	AUC	SEN	SPE
AD vs HC	76%	0.76	83%	78%
pMCI vs sMCI	77%	0.78	88%	69%

Tabla E.10: Métricas obtenidas por el experimento 10.

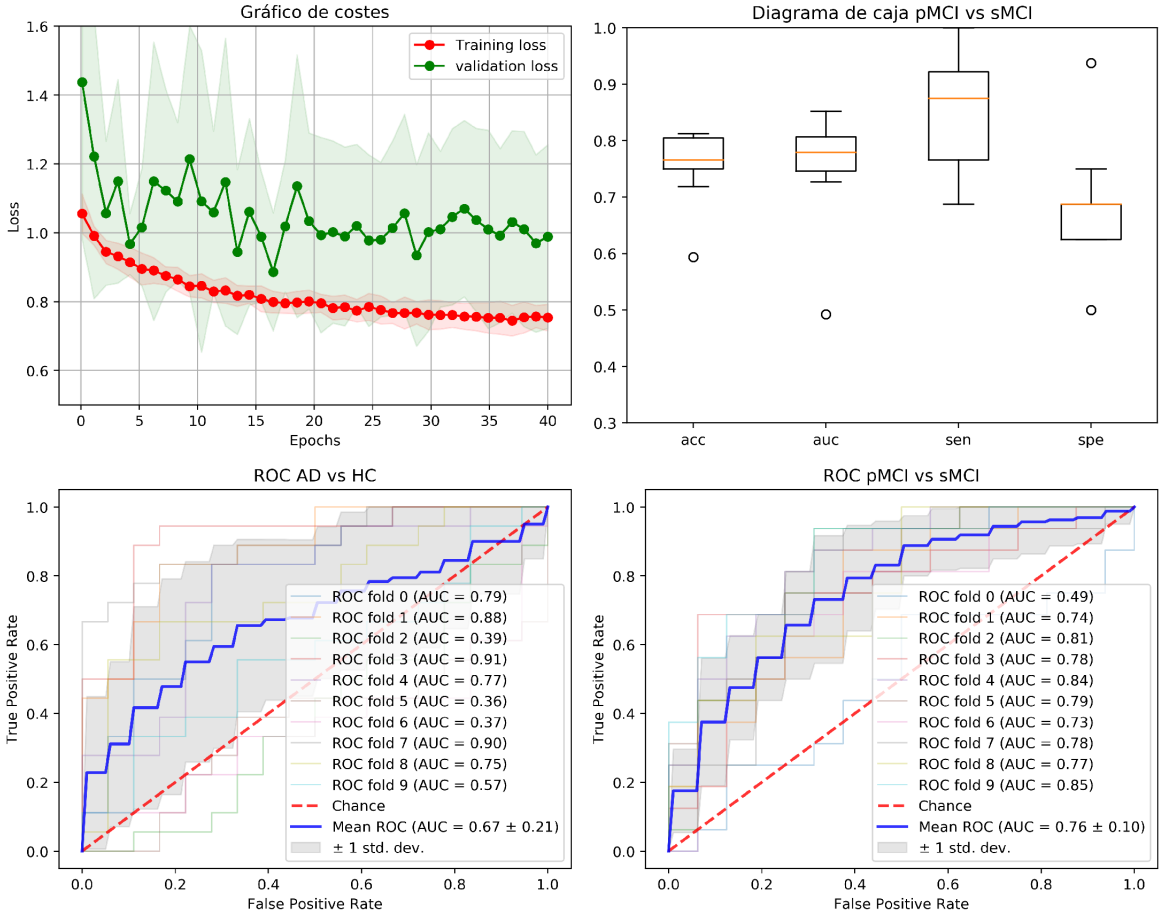


Figura E.10: Gráficas obtenidas por el experimento 10. En la esquina superior izquierda un gráfico de convergencia de la función de coste, en la esquina superior derecha un diagrama de cajas para el problema pMCI vs sMCI, en la esquina inferior izquierda una curva ROC para el problema de AD vs HC y en la esquina inferior derecha una curva ROC para el problema de pMCI vs sMCI.

Experimento *11* : ANTS,SSD,*skull stripping*

	ACC	AUC	SEN	SPE
AD vs HC	78%	0.83	86%	67%
pMCI vs sMCI	72%	0.71	72%	75%

Tabla E.11: Métricas obtenidas por el experimento 11.

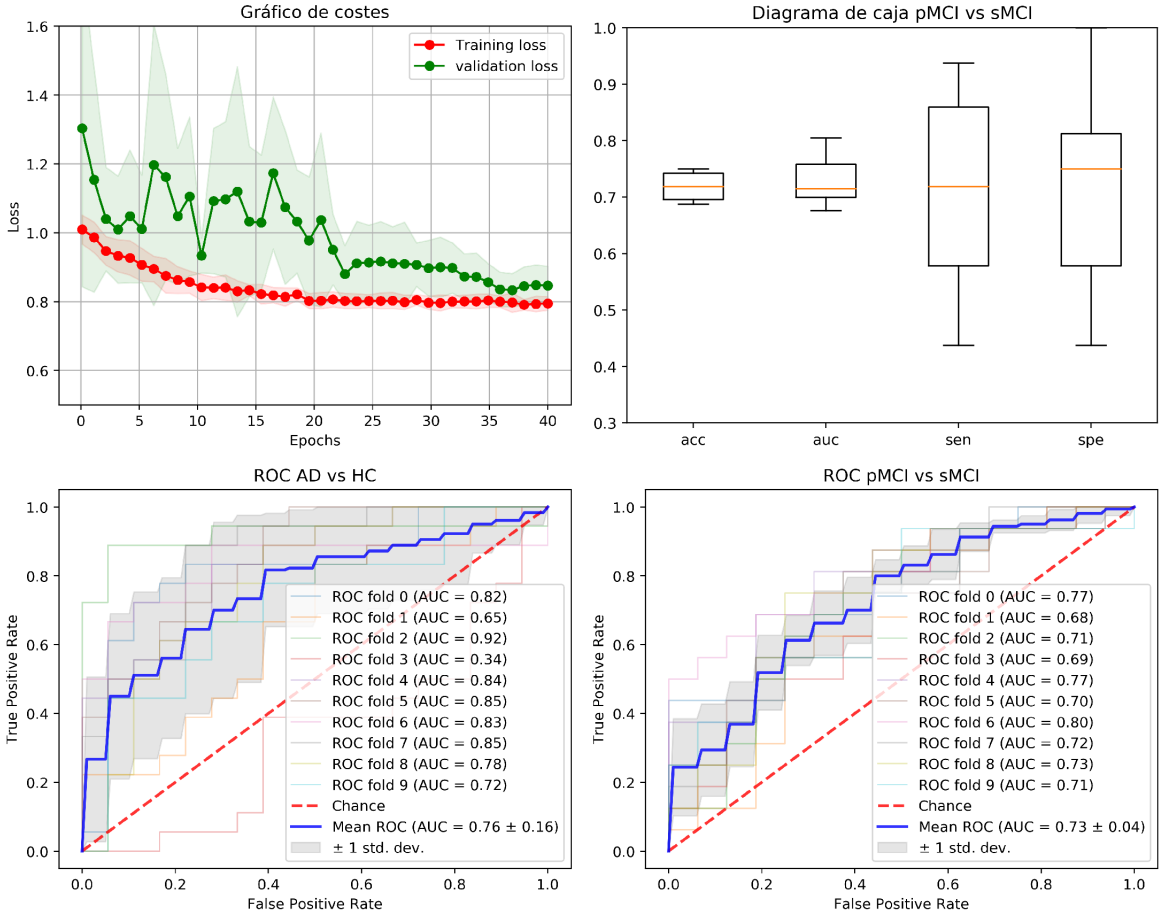


Figura E.11: Gráficas obtenidas por el experimento 11. En la esquina superior izquierda un gráfico de convergencia de la función de coste, en la esquina superior derecha un diagrama de cajas para el problema pMCI vs sMCI, en la esquina inferior izquierda una curva ROC para el problema de AD vs HC y en la esquina inferior derecha una curva ROC para el problema de pMCI vs sMCI.

Experimento *12 :* ANTS,SSD,*skull stripping*

Con estandarización por característica.

	ACC	AUC	SEN	SPE
AD vs HC	61%	0.57	89%	33%
pMCI vs sMCI	69%	0.68	81%	62%

Tabla E.12: Métricas obtenidas por el experimento 12.

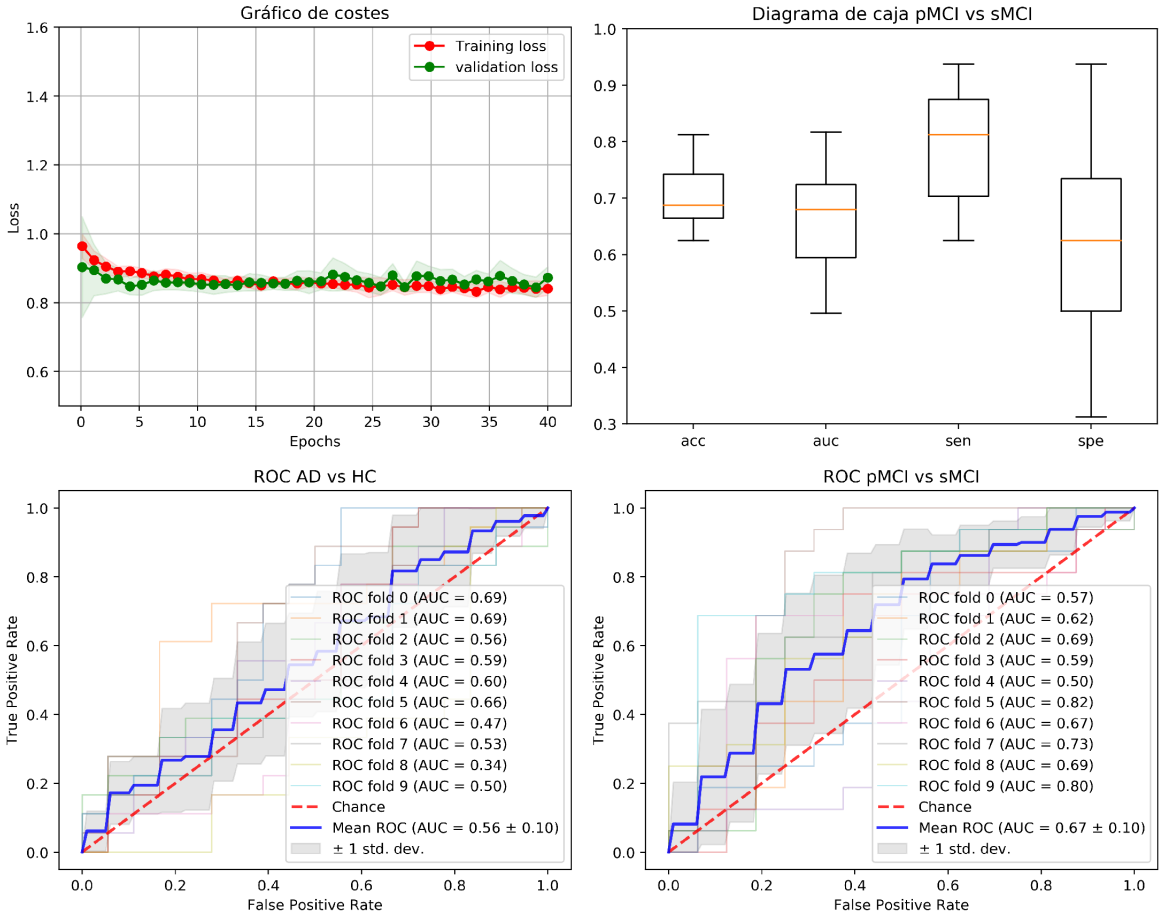


Figura E.12: Gráficas obtenidas por el experimento 12. En la esquina superior izquierda un gráfico de convergencia de la función de coste, en la esquina superior derecha un diagrama de cajas para el problema pMCI vs sMCI, en la esquina inferior izquierda una curva ROC para el problema de AD vs HC y en la esquina inferior derecha una curva ROC para el problema de pMCI vs sMCI.

Experimento *13 :* BL PDE-LDDMM, SSD,*skull stripping*:

	ACC	AUC	SEN	SPE
AD vs HC	69%	0.73	61%	86%
pMCI vs sMCI	70%	0.67	66%	72%

Tabla E.13: Métricas obtenidas por el experimento 13.

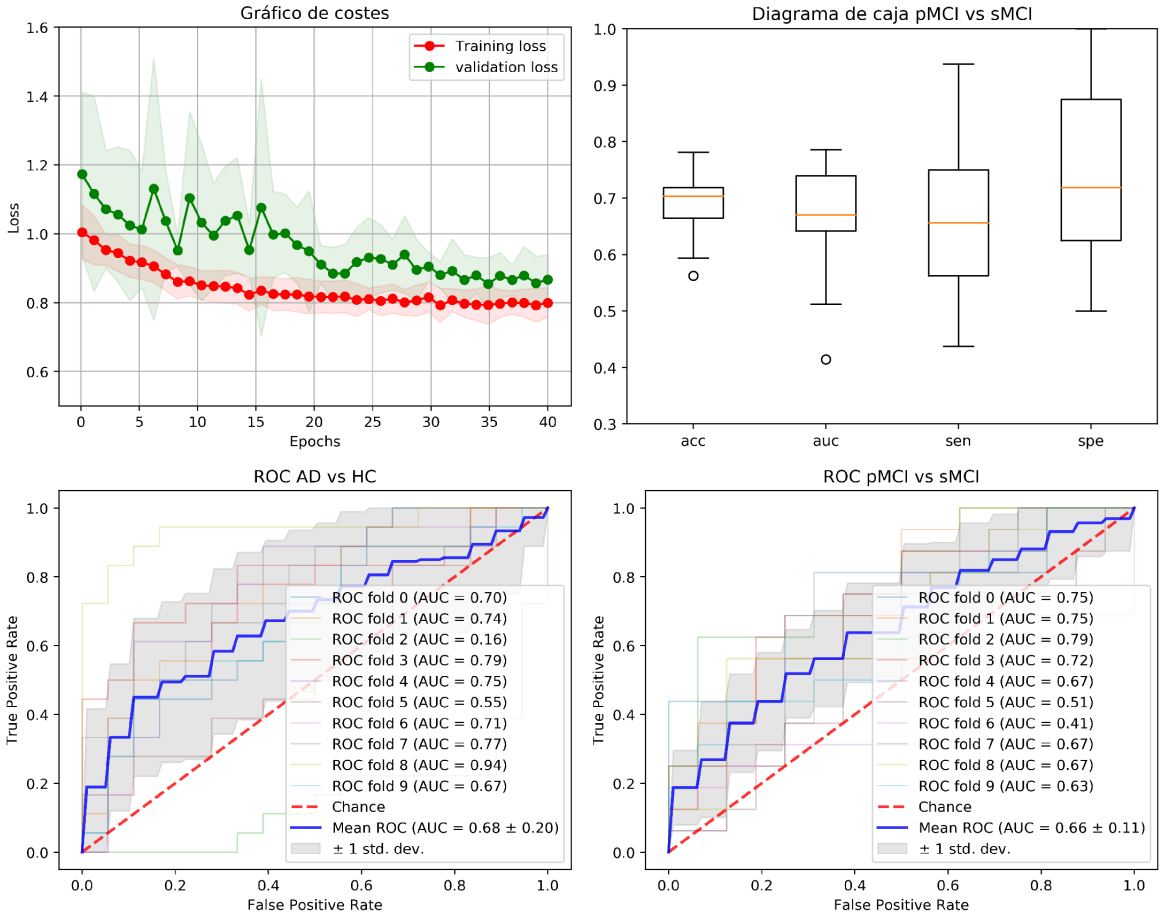


Figura E.13: Gráficas obtenidas por el experimento 13. En la esquina superior izquierda un gráfico de convergencia de la función de coste, en la esquina superior derecha un diagrama de cajas para el problema pMCI vs sMCI, en la esquina inferior izquierda una curva ROC para el problema de AD vs HC y en la esquina inferior derecha una curva ROC para el problema de pMCI vs sMCI.

Datos de entrada: Clínico+MRI

Experimento *14 :* Sin registro

	ACC	AUC	SEN	SPE
AD vs HC	100%	1.00	100%	100%
pMCI vs sMCI	88%	0.91	88%	94%

Tabla E.14: Métricas obtenidas por el experimento 14.

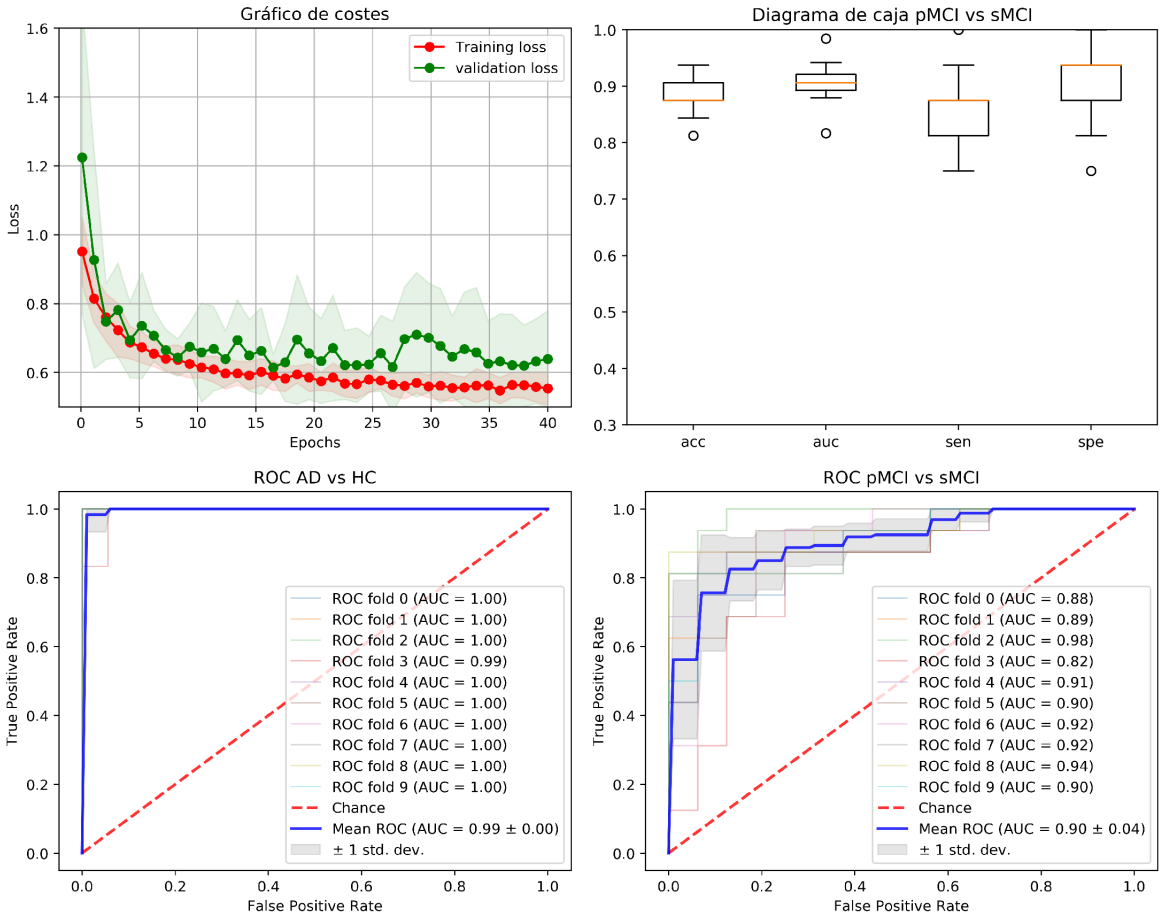


Figura E.14: Gráficas obtenidas por el experimento 14. En la esquina superior izquierda un gráfico de convergencia de la función de coste, en la esquina superior derecha un diagrama de cajas para el problema pMCI vs sMCI, en la esquina inferior izquierda una curva ROC para el problema de AD vs HC y en la esquina inferior derecha una curva ROC para el problema de pMCI vs sMCI.

Experimento *15* : Sin registro,*skull stripping*

	ACC	AUC	SEN	SPE
AD vs HC	100%	1.00	100%	100%
pMCI vs sMCI	88%	0.92	81%	88%

Tabla E.15: Métricas obtenidas por el experimento 15.

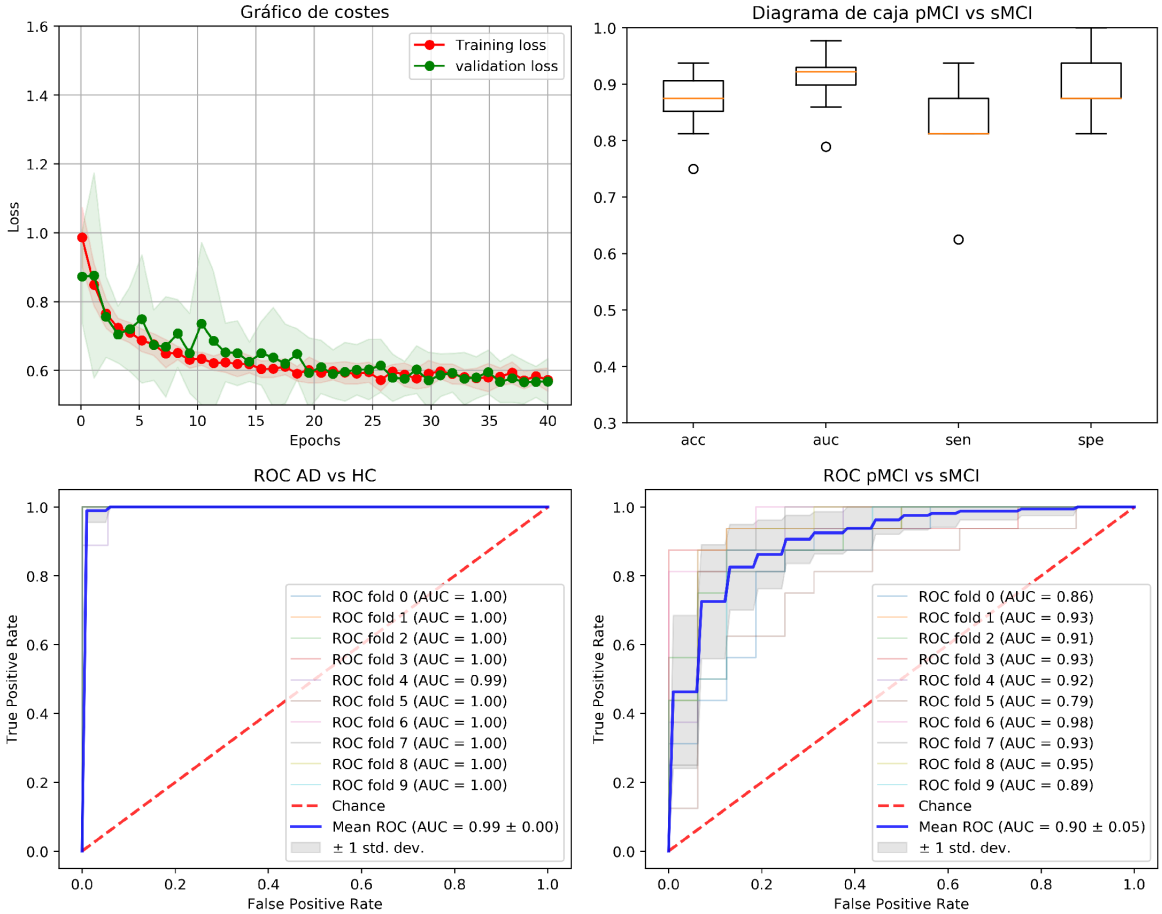


Figura E.15: Gráficas obtenidas por el experimento 15. En la esquina superior izquierda un gráfico de convergencia de la función de coste, en la esquina superior derecha un diagrama de cajas para el problema pMCI vs sMCI, en la esquina inferior izquierda una curva ROC para el problema de AD vs HC y en la esquina inferior derecha una curva ROC para el problema de pMCI vs sMCI.

Experimento *16* : *Affine registration,skull stripping*

	ACC	AUC	SEN	SPE
AD vs HC	100%	1.00	100%	100%
pMCI vs sMCI	89%	0.94	88%	94%

Tabla E.16: Métricas obtenidas por el experimento 16.

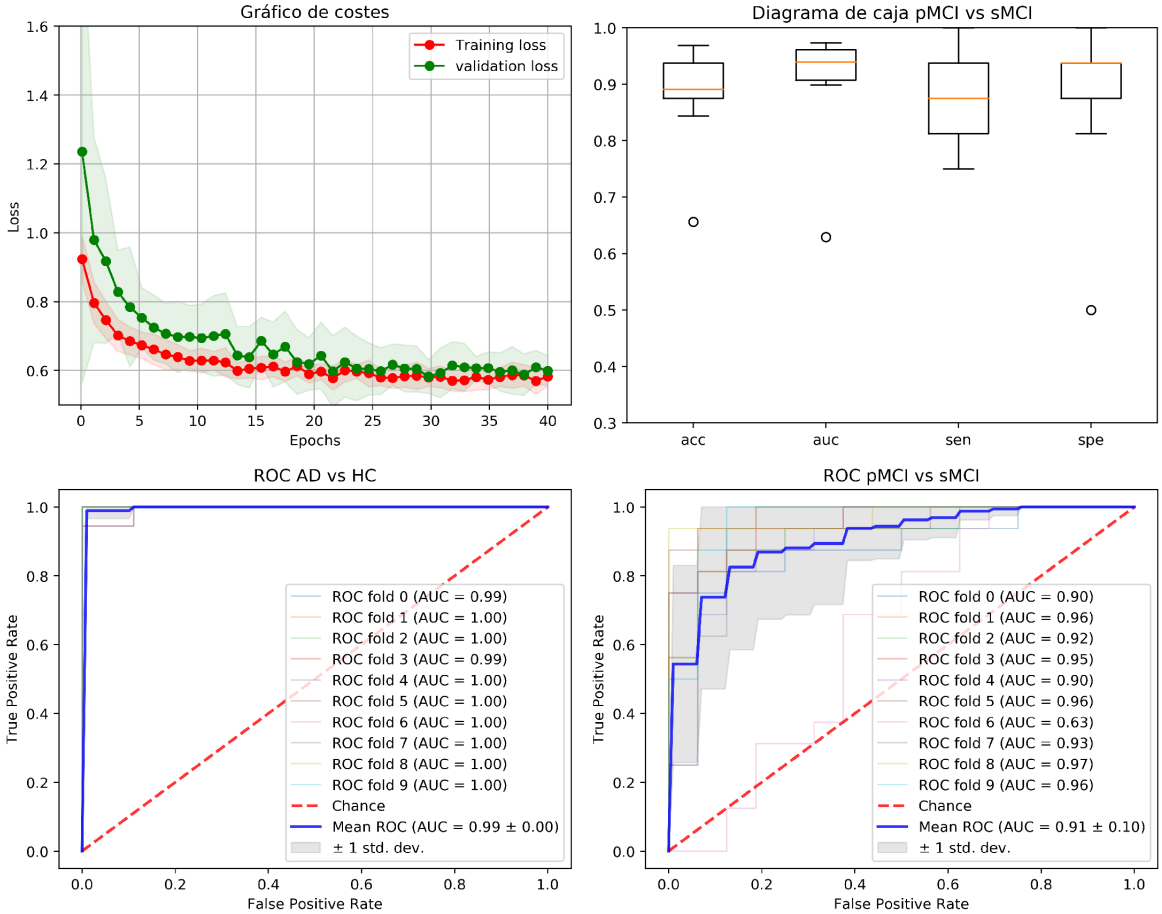


Figura E.16: Gráficas obtenidas por el experimento 16. En la esquina superior izquierda un gráfico de convergencia de la función de coste, en la esquina superior derecha un diagrama de cajas para el problema pMCI vs sMCI, en la esquina inferior izquierda una curva ROC para el problema de AD vs HC y en la esquina inferior derecha una curva ROC para el problema de pMCI vs sMCI.

Experimento *17* : *Affine registration,skull stripping*

Con estandarización por muestra y luego normalización por característica.

	ACC	AUC	SEN	SPE
AD vs HC	99%	1.00	100%	100%
pMCI vs sMCI	88%	0.92	94%	88%

Tabla E.17: Métricas obtenidas por el experimento 17.

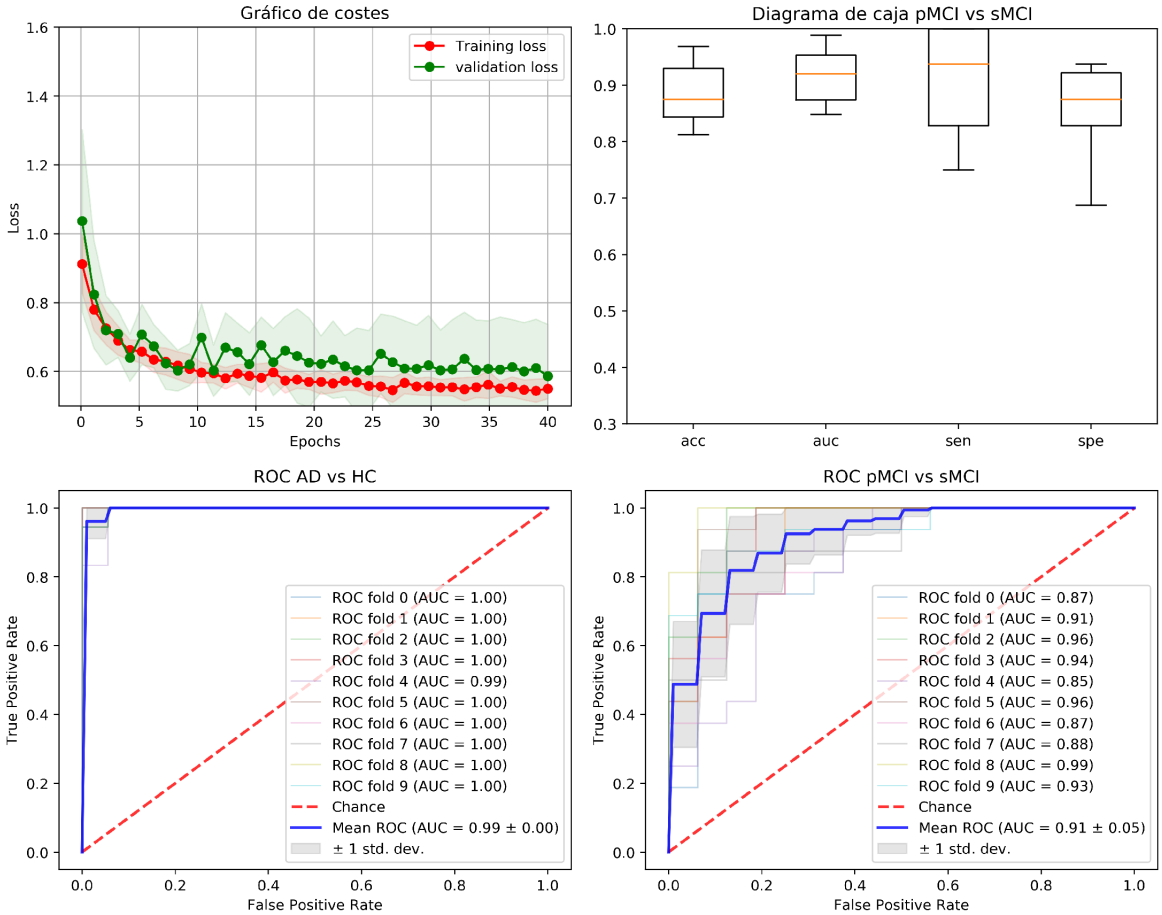


Figura E.17: Gráficas obtenidas por el experimento 17. En la esquina superior izquierda un gráfico de convergencia de la función de coste, en la esquina superior derecha un diagrama de cajas para el problema pMCI vs sMCI, en la esquina inferior izquierda una curva ROC para el problema de AD vs HC y en la esquina inferior derecha una curva ROC para el problema de pMCI vs sMCI.

Experimento *18* : *Affine registration,skull stripping*

Con estandarización por muestra y luego normalización por característica.

	ACC	AUC	SEN	SPE
AD vs HC	100%	1.00	100%	100%
pMCI vs sMCI	88%	0.89	88%	88%

Tabla E.18: Métricas obtenidas por el experimento 18.

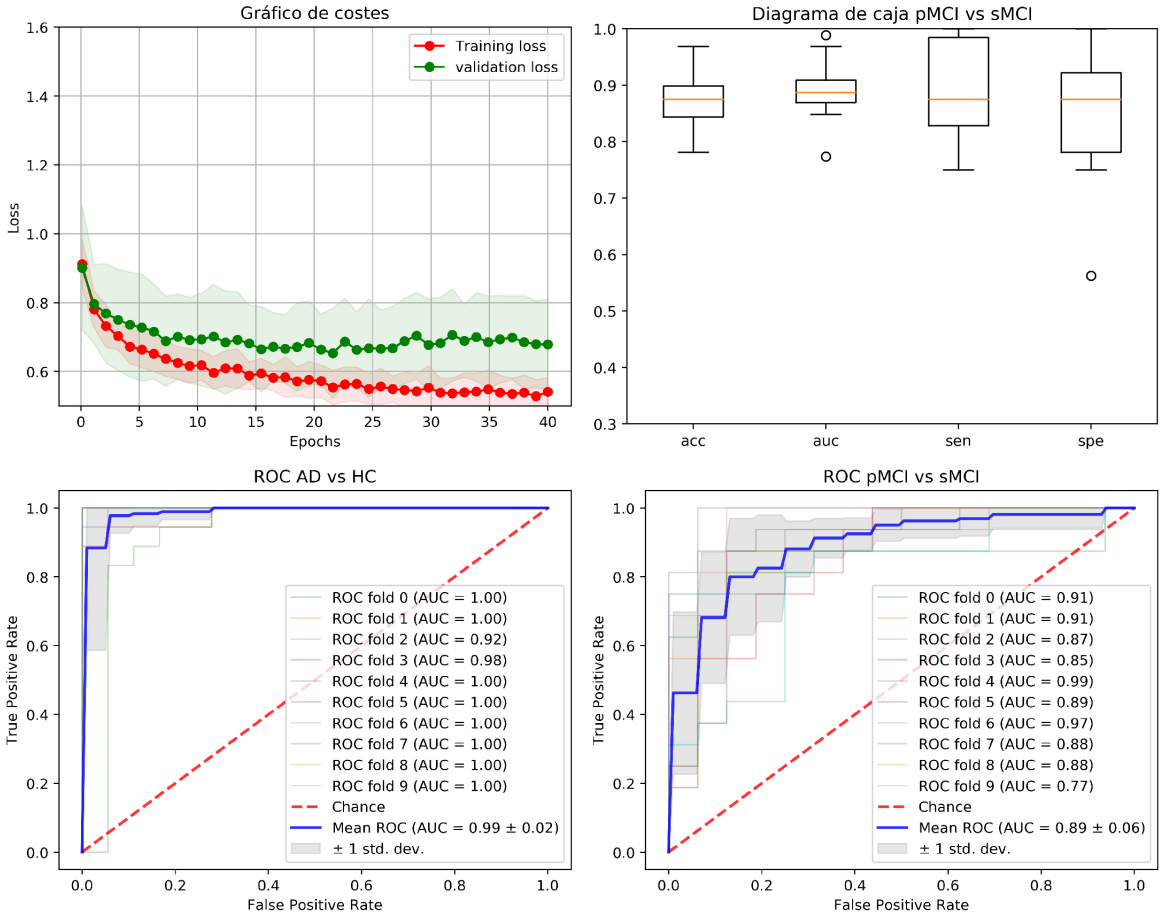


Figura E.18: Gráficas obtenidas por el experimento 18. En la esquina superior izquierda un gráfico de convergencia de la función de coste, en la esquina superior derecha un diagrama de cajas para el problema pMCI vs sMCI, en la esquina inferior izquierda una curva ROC para el problema de AD vs HC y en la esquina inferior derecha una curva ROC para el problema de pMCI vs sMCI.

Experimento *19* : *Affine registration,skull stripping*

Con estandarización por característica y mascara mínima tras registro.

	ACC	AUC	SEN	SPE
AD vs HC	100%	1.00	100%	100%
pMCI vs sMCI	84%	0.90	88%	88%

Tabla E.19: Métricas obtenidas por el experimento 19.

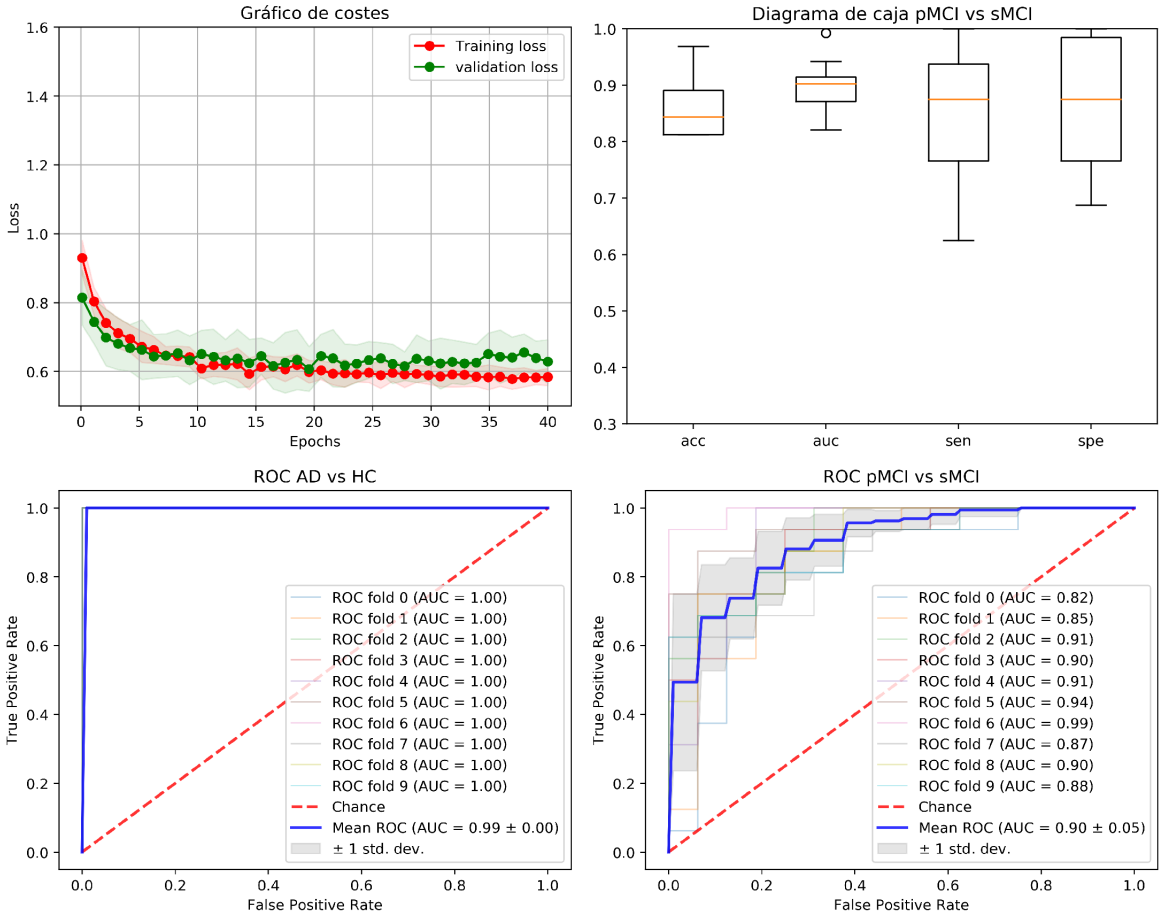


Figura E.19: Gráficas obtenidas por el experimento 19. En la esquina superior izquierda un gráfico de convergencia de la función de coste, en la esquina superior derecha un diagrama de cajas para el problema pMCI vs sMCI, en la esquina inferior izquierda una curva ROC para el problema de AD vs HC y en la esquina inferior derecha una curva ROC para el problema de pMCI vs sMCI.

Experimento *20 :* BL PDE-LDDMM, NCC,*skull stripping*

	ACC	AUC	SEN	SPE
AD vs HC	100%	1.00	100%	100%
pMCI vs sMCI	89%	0.94	94%	91%

Tabla E.20: Métricas obtenidas por el experimento 20.

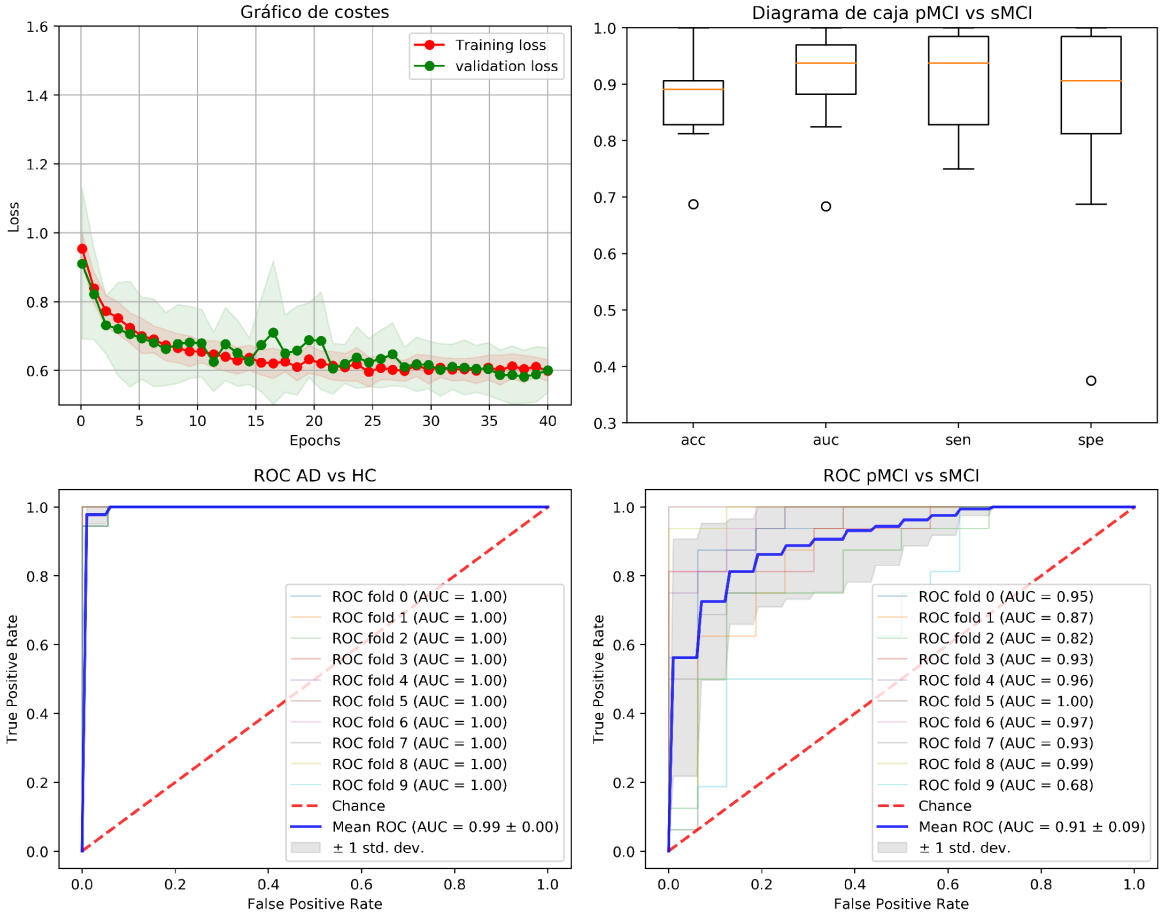


Figura E.20: Gráficas obtenidas por el experimento 20. En la esquina superior izquierda un gráfico de convergencia de la función de coste, en la esquina superior derecha un diagrama de cajas para el problema pMCI vs sMCI, en la esquina inferior izquierda una curva ROC para el problema de AD vs HC y en la esquina inferior derecha una curva ROC para el problema de pMCI vs sMCI.

Experimento *21 :* ANTS,SSD

Con estandarización por característica.

	ACC	AUC	SEN	SPE
AD vs HC	100%	1.00	100%	100%
pMCI vs sMCI	86%	0.88	88%	81%

Tabla E.21: Métricas obtenidas por el experimento 21.

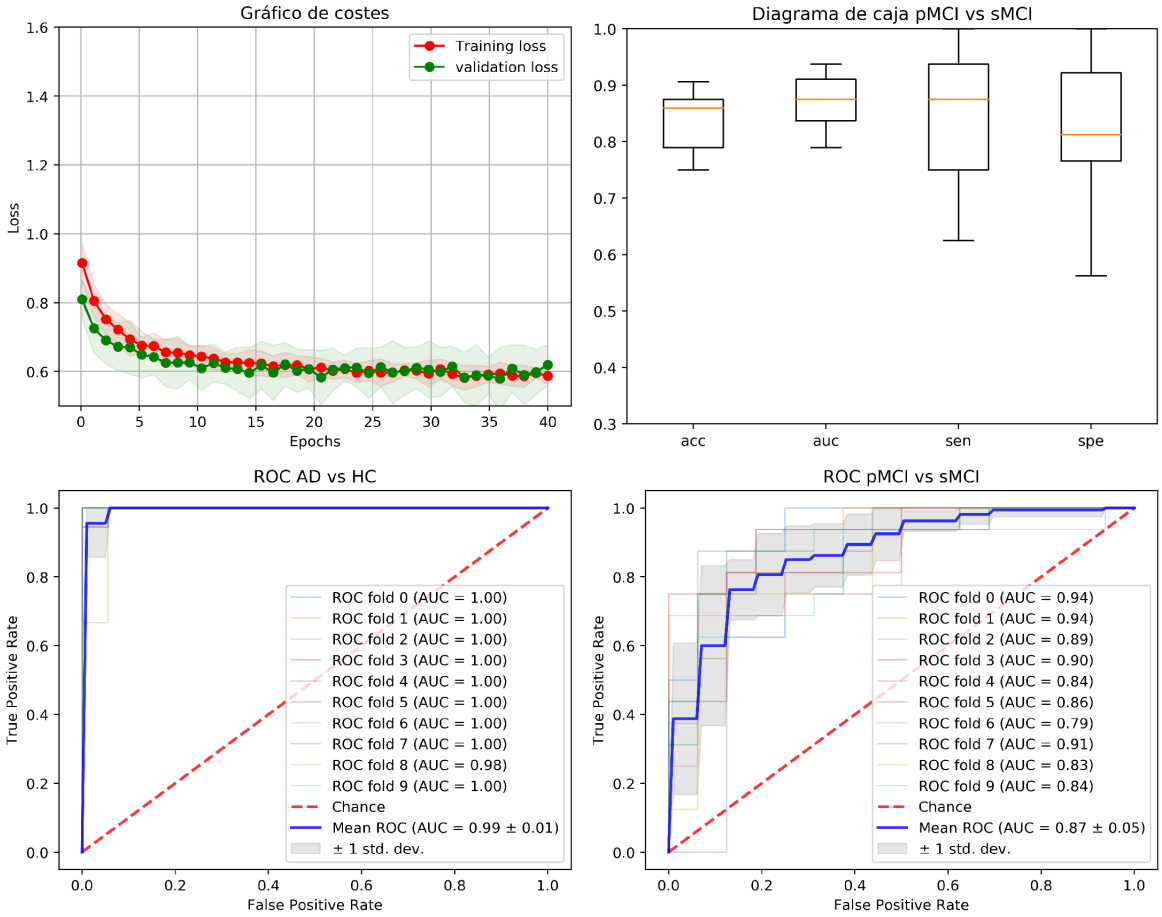


Figura E.21: Gráficas obtenidas por el experimento 21. En la esquina superior izquierda un gráfico de convergencia de la función de coste, en la esquina superior derecha un diagrama de cajas para el problema pMCI vs sMCI, en la esquina inferior izquierda una curva ROC para el problema de AD vs HC y en la esquina inferior derecha una curva ROC para el problema de pMCI vs sMCI.

Experimento *22 :* ANTS,SSD

	ACC	AUC	SEN	SPE
AD vs HC	100%	1.00	100%	100%
pMCI vs sMCI	86%	0.92	81%	94%

Tabla E.22: Métricas obtenidas por el experimento 22.

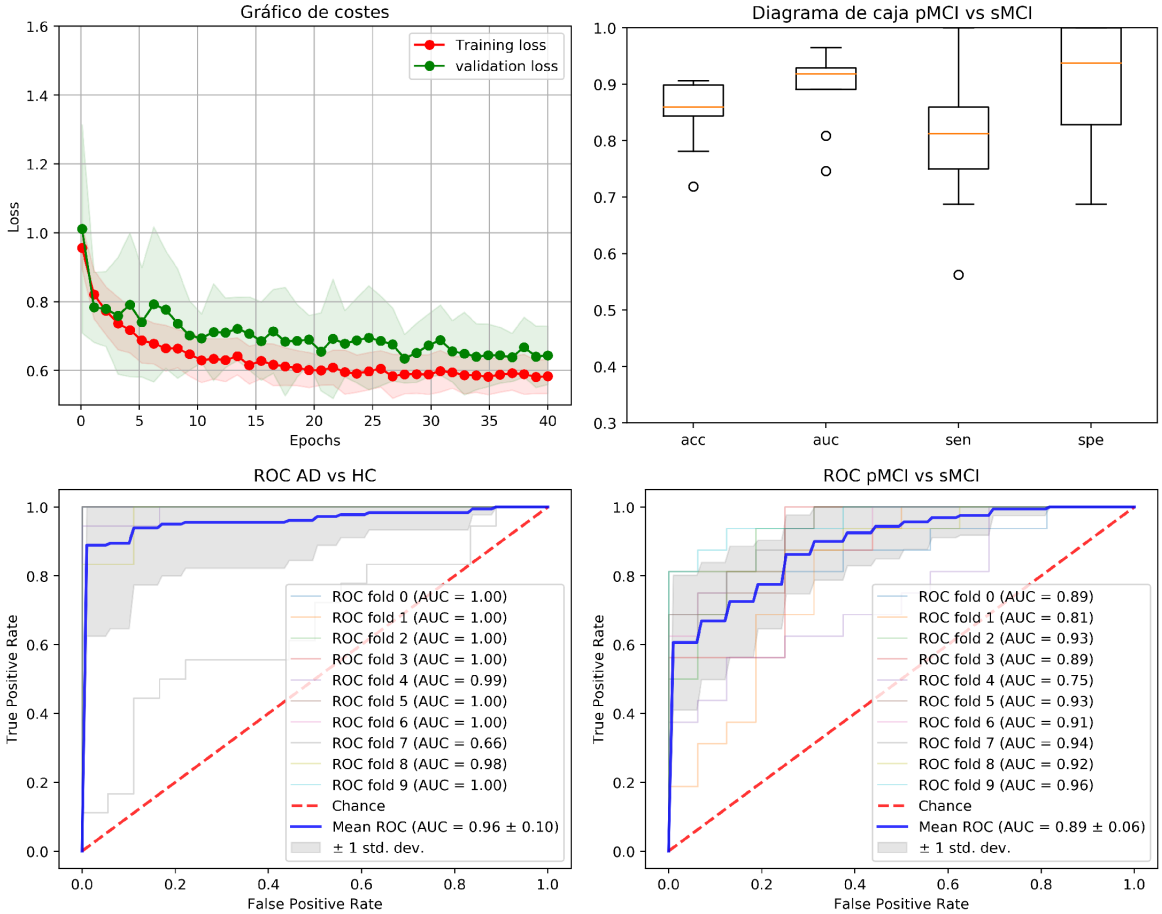


Figura E.22: Gráficas obtenidas por el experimento 22. En la esquina superior izquierda un gráfico de convergencia de la función de coste, en la esquina superior derecha un diagrama de cajas para el problema pMCI vs sMCI, en la esquina inferior izquierda una curva ROC para el problema de AD vs HC y en la esquina inferior derecha una curva ROC para el problema de pMCI vs sMCI.

Experimento *23 :* ANTS,SSD,skull stripping

	ACC	AUC	SEN	SPE
AD vs HC	100%	1.00	100%	100%
pMCI vs sMCI	86%	0.90	84%	88%

Tabla E.23: Métricas obtenidas por el experimento 23.

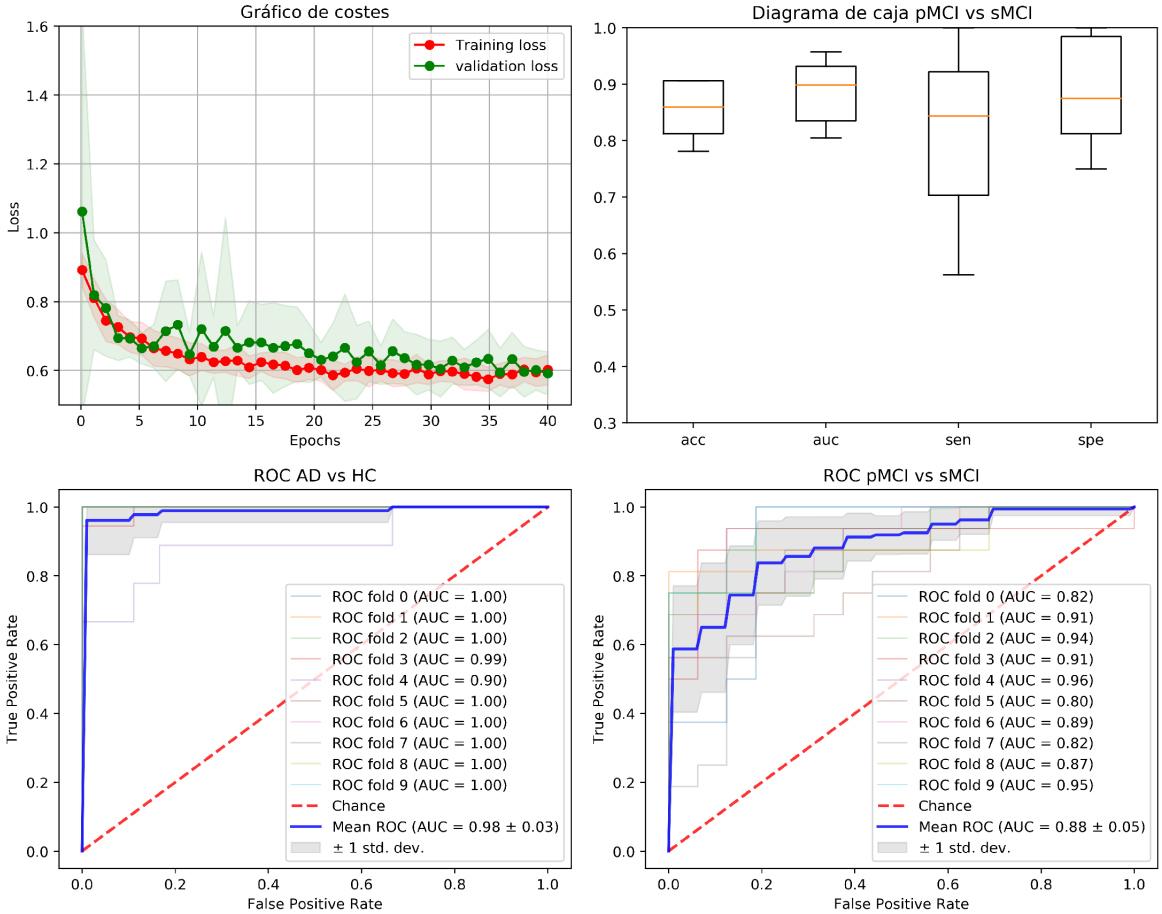


Figura E.23: Gráficas obtenidas por el experimento 23. En la esquina superior izquierda un gráfico de convergencia de la función de coste, en la esquina superior derecha un diagrama de cajas para el problema pMCI vs sMCI, en la esquina inferior izquierda una curva ROC para el problema de AD vs HC y en la esquina inferior derecha una curva ROC para el problema de pMCI vs sMCI.

Experimento *24 :* ANTS,SSD,skull stripping

Con estandarización por característica.

	ACC	AUC	SEN	SPE
AD vs HC	100%	1.00	100%	100%
pMCI vs sMCI	86%	0.91	84%	81%

Tabla E.24: Métricas obtenidas por el experimento 24.

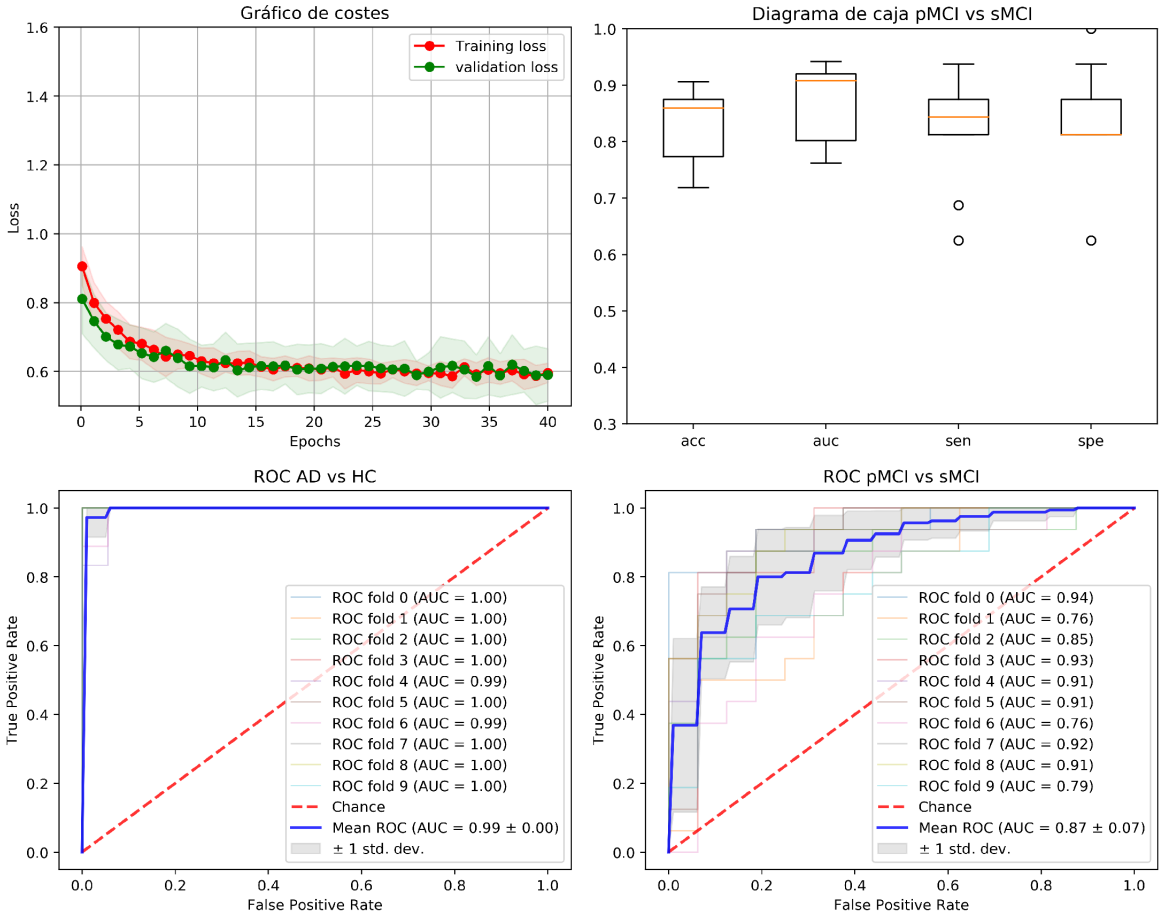


Figura E.24: Gráficas obtenidas por el experimento 24. En la esquina superior izquierda un gráfico de convergencia de la función de coste, en la esquina superior derecha un diagrama de cajas para el problema pMCI vs sMCI, en la esquina inferior izquierda una curva ROC para el problema de AD vs HC y en la esquina inferior derecha una curva ROC para el problema de pMCI vs sMCI.

Experimento *25 :* BL PDE-LDDMM, SSD,skull stripping

	ACC	AUC	SEN	SPE
AD vs HC	100%	1.00	100%	100%
pMCI vs sMCI	88%	0.90	97%	81%

Tabla E.25: Métricas obtenidas por el experimento 25.

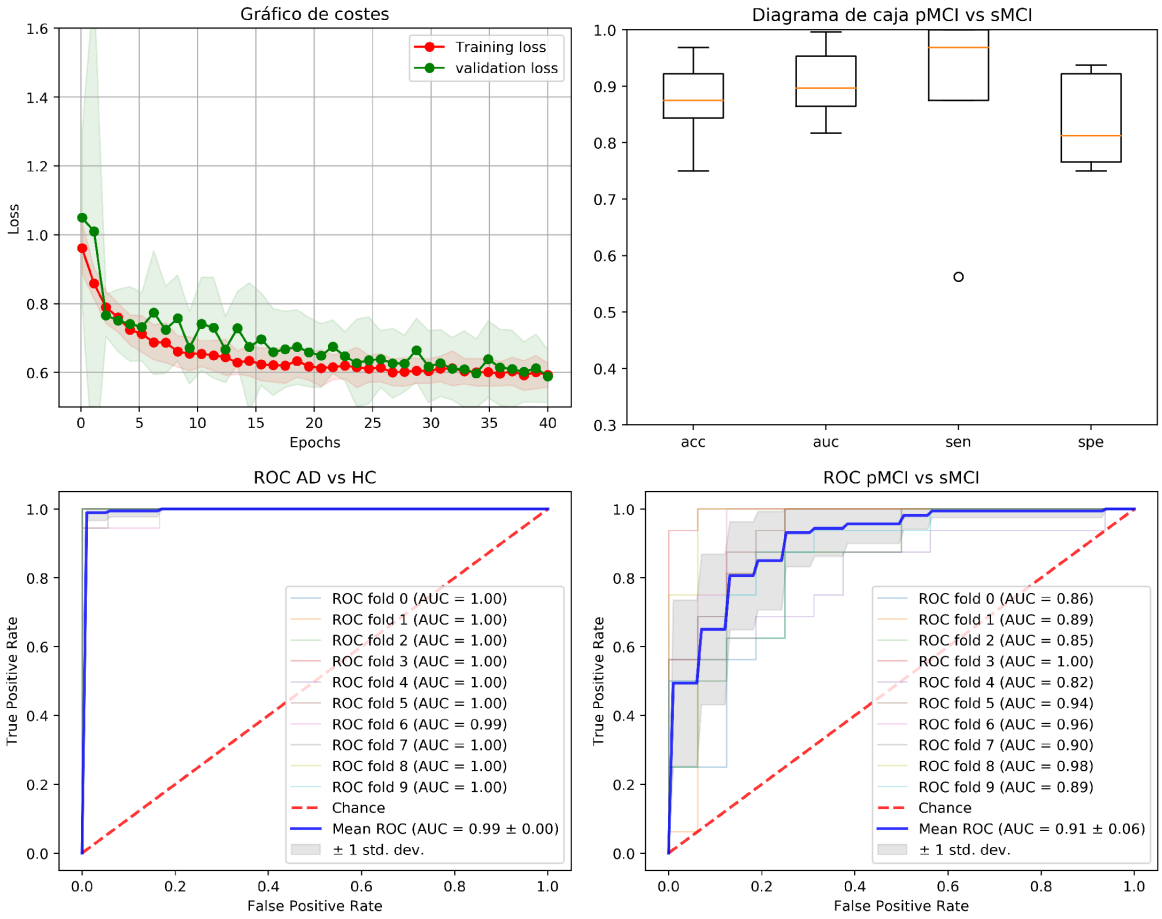


Figura E.25: Gráficas obtenidas por el experimento 25. En la esquina superior izquierda un gráfico de convergencia de la función de coste, en la esquina superior derecha un diagrama de cajas para el problema pMCI vs sMCI, en la esquina inferior izquierda una curva ROC para el problema de AD vs HC y en la esquina inferior derecha una curva ROC para el problema de pMCI vs sMCI.

Datos de entrada: Clínico+MRI+Jac

Experimento *26 :* BL PDE-LDDMM, NCC,skull stripping

	ACC	AUC	SEN	SPE
AD vs HC	100%	1.00	100%	100%
pMCI vs sMCI	88%	0.91	88%	88%

Tabla E.26: Métricas obtenidas por el experimento 26.

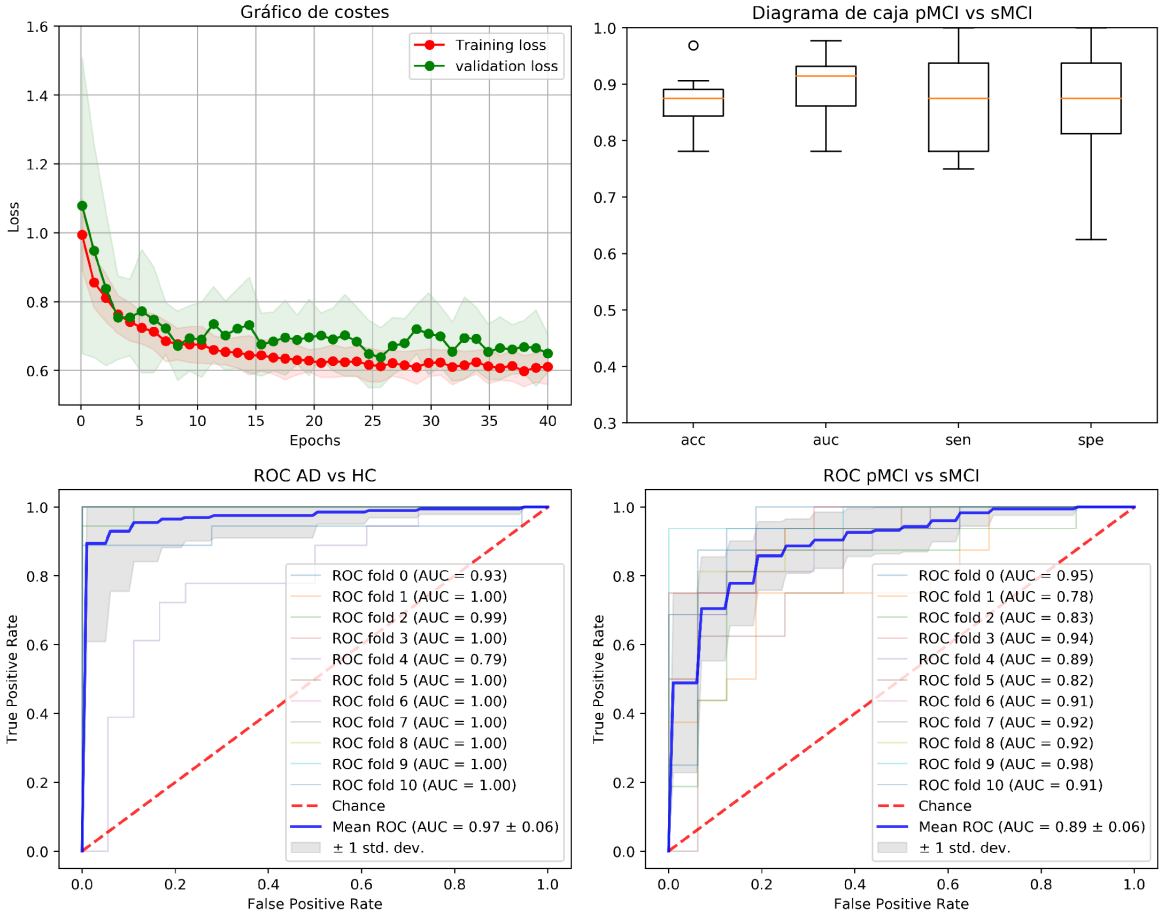


Figura E.26: Gráficas obtenidas por el experimento 26. En la esquina superior izquierda un gráfico de convergencia de la función de coste, en la esquina superior derecha un diagrama de cajas para el problema pMCI vs sMCI, en la esquina inferior izquierda una curva ROC para el problema de AD vs HC y en la esquina inferior derecha una curva ROC para el problema de pMCI vs sMCI.

Experimento *27 :* ANTS,SSD

	ACC	AUC	SEN	SPE
AD vs HC	100%	1.00	100%	100%
pMCI vs sMCI	88%	0.92	94%	84%

Tabla E.27: Métricas obtenidas por el experimento 27.

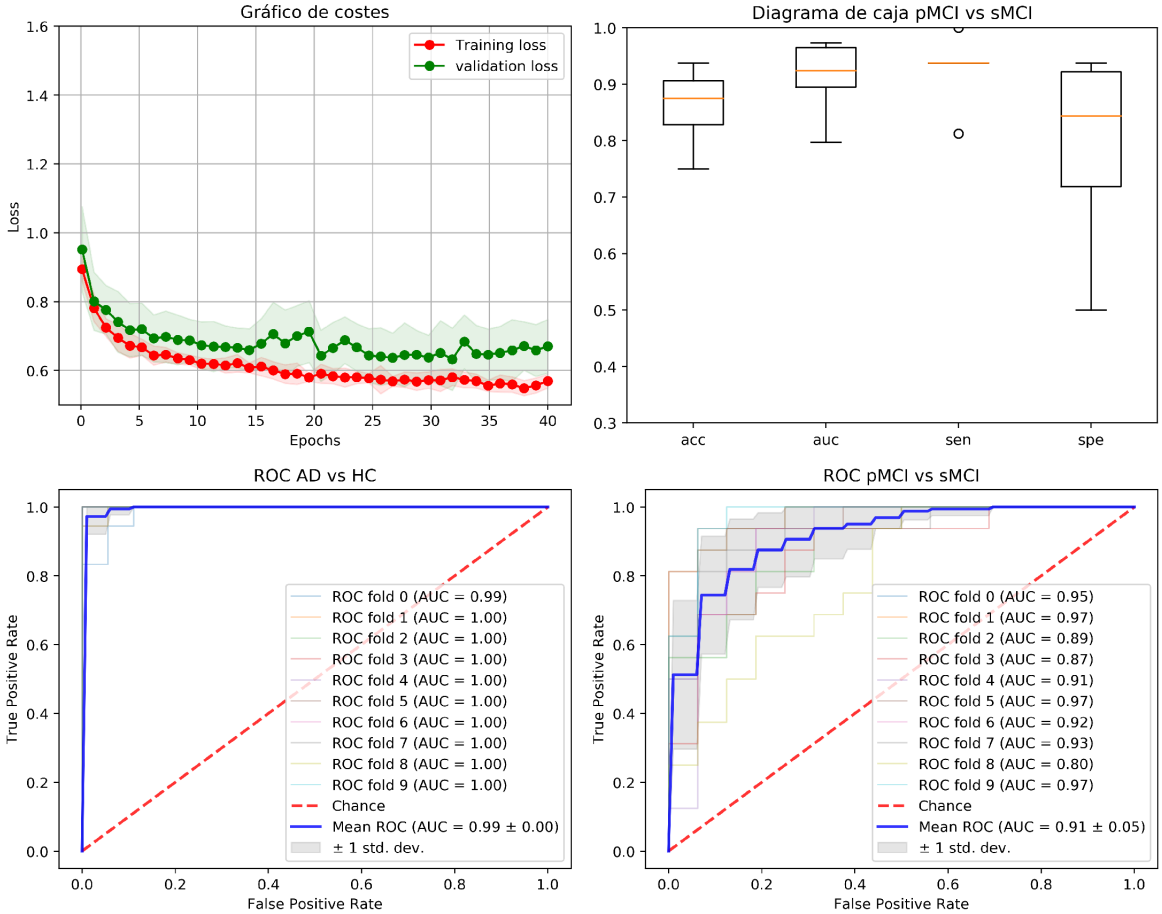


Figura E.27: Gráficas obtenidas por el experimento 27. En la esquina superior izquierda un gráfico de convergencia de la función de coste, en la esquina superior derecha un diagrama de cajas para el problema pMCI vs sMCI, en la esquina inferior izquierda una curva ROC para el problema de AD vs HC y en la esquina inferior derecha una curva ROC para el problema de pMCI vs sMCI.

Experimento *28 :* ANTS,SSD,skull stripping

	ACC	AUC	SEN	SPE
AD vs HC	100%	1.00	100%	100%
pMCI vs sMCI	88%	0.90	81%	84%

Tabla E.28: Métricas obtenidas por el experimento 28.

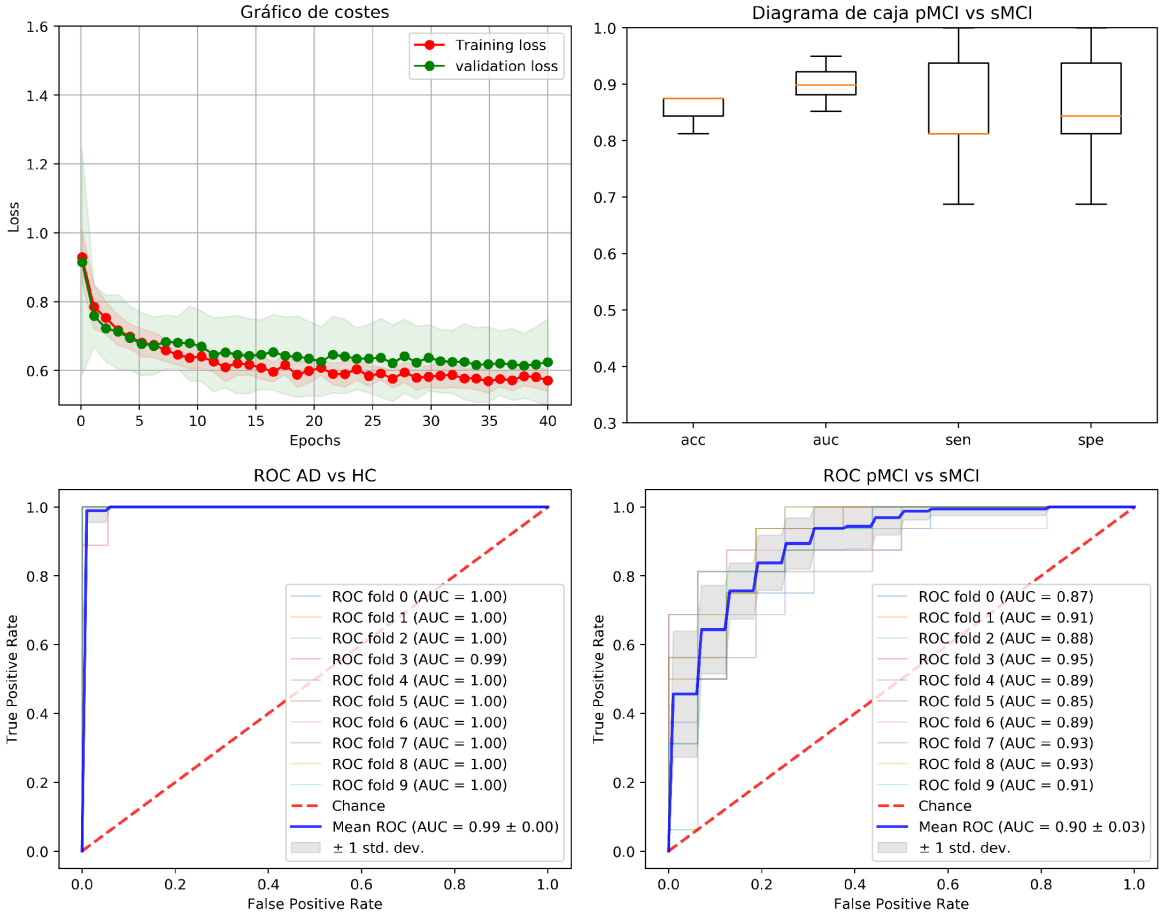


Figura E.28: Gráficas obtenidas por el experimento 28. En la esquina superior izquierda un gráfico de convergencia de la función de coste, en la esquina superior derecha un diagrama de cajas para el problema pMCI vs sMCI, en la esquina inferior izquierda una curva ROC para el problema de AD vs HC y en la esquina inferior derecha una curva ROC para el problema de pMCI vs sMCI.

Experimento *29 :* ANTS,SSD,skull stripping

Con estandarización por característica.

	ACC	AUC	SEN	SPE
AD vs HC	100%	1.00	100%	100%
pMCI vs sMCI	84%	0.89	88%	81%

Tabla E.29: Métricas obtenidas por el experimento 29.

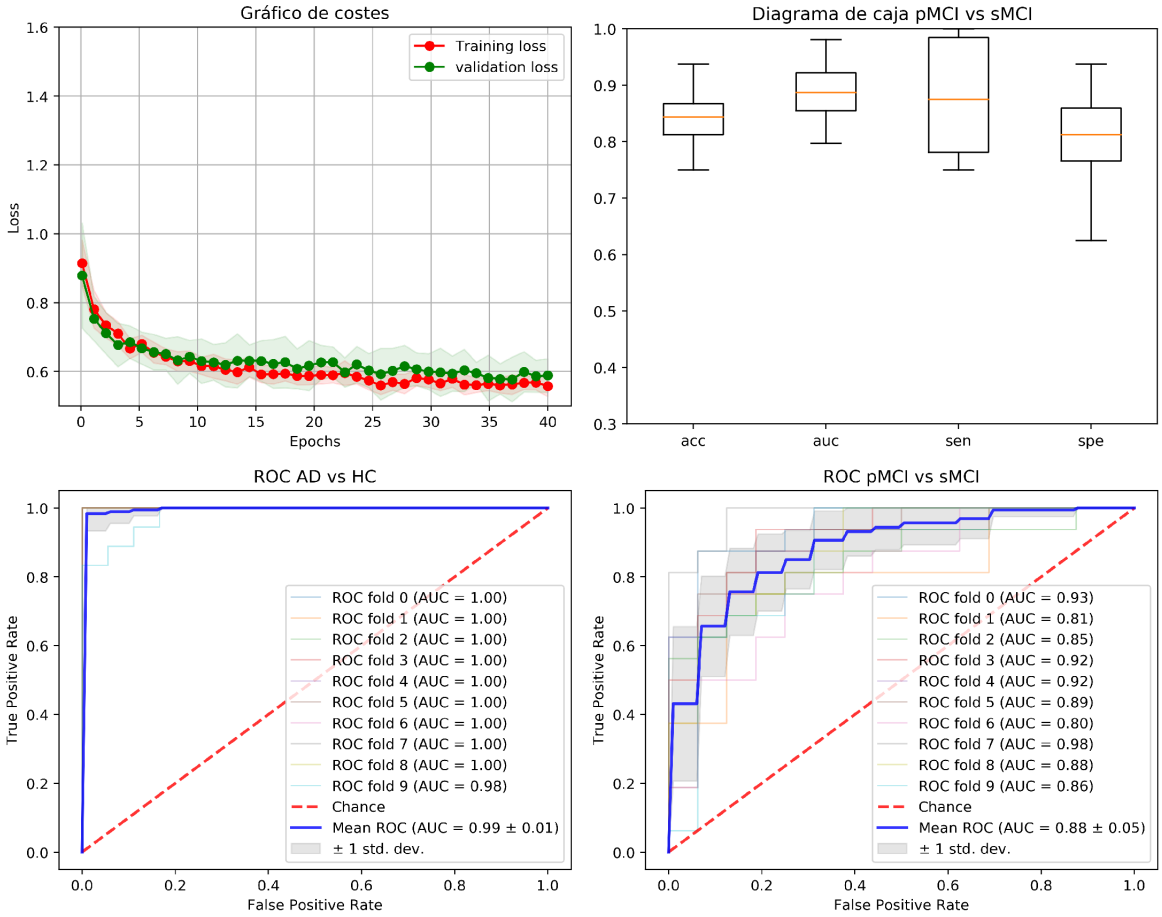


Figura E.29: Gráficas obtenidas por el experimento 29. En la esquina superior izquierda un gráfico de convergencia de la función de coste, en la esquina superior derecha un diagrama de cajas para el problema pMCI vs sMCI, en la esquina inferior izquierda una curva ROC para el problema de AD vs HC y en la esquina inferior derecha una curva ROC para el problema de pMCI vs sMCI.

Experimento *30 :* BL PDE-LDDMM, SSD,skull stripping

	ACC	AUC	SEN	SPE
AD vs HC	100%	1.00	100%	100%
pMCI vs sMCI	88%	0.90	84%	88%

Tabla E.30: Métricas obtenidas por el experimento 30.

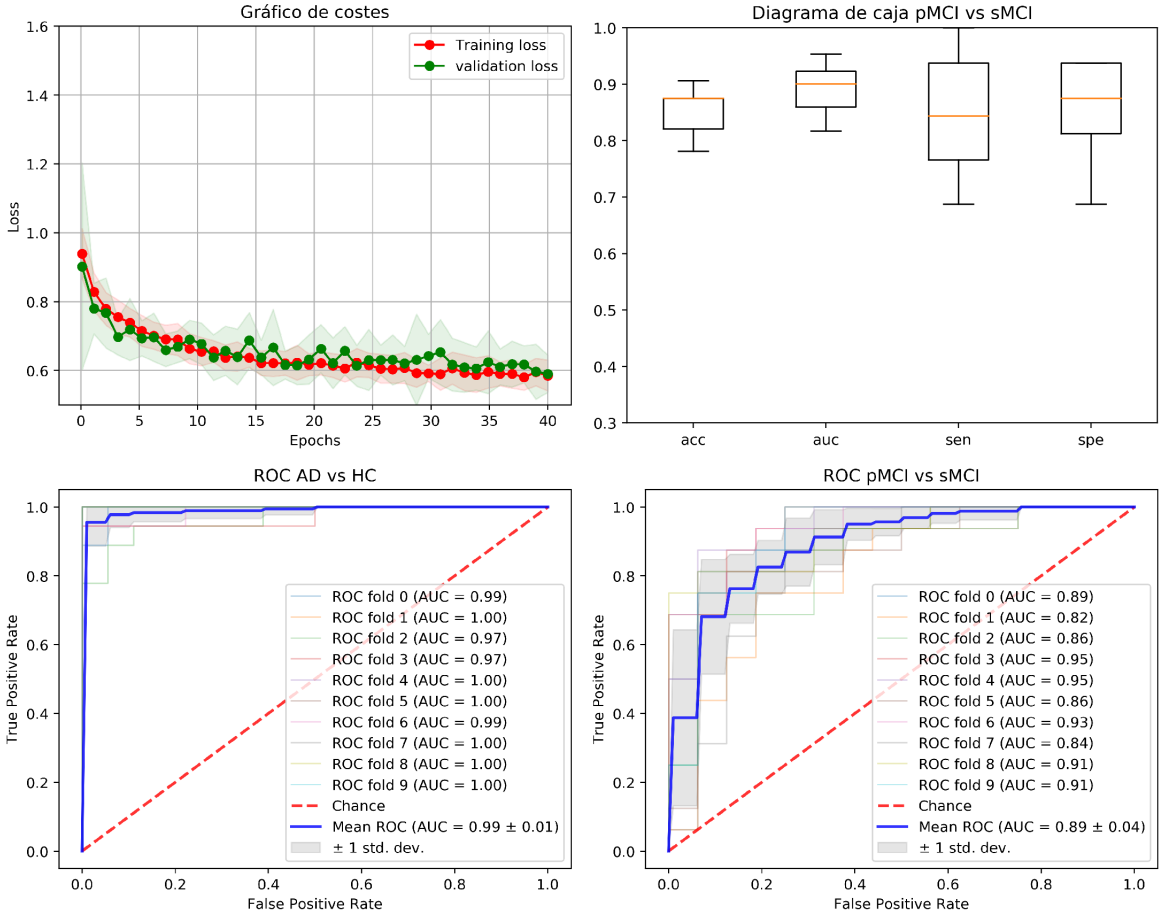


Figura E.30: Gráficas obtenidas por el experimento 30. En la esquina superior izquierda un gráfico de convergencia de la función de coste, en la esquina superior derecha un diagrama de cajas para el problema pMCI vs sMCI, en la esquina inferior izquierda una curva ROC para el problema de AD vs HC y en la esquina inferior derecha una curva ROC para el problema de pMCI vs sMCI.